Trabajo fin de grado

Grado en Ingeniería en Tecnologías Industriales

Time series data augmentation

MEMORIA

Autor: Xavier Miralbell Bes
Director: Samir Kanaan

Convocatoria: Julio 2021

Escola Tècnica Superior
d'Enginyeria Industrial de Barcelona

Abstract

Data augmentation is a very powerful tool to increase the size of the training set and, therefore, to improve the learning process of neural networks.

It is a well-studied technique in imaging, but it is not as well developed for other types of data such as time series.

For this reason, this work sets as its main objective the data augmentation of temporal data sets, for which two sub-objectives were obtained: on the one hand, to analyze the data of the time series and its variables, and, on the other, the study and the creation of generative models.

To achieve these objectives, a specific data set is used with which we will work using the tools provided by Python and the Pandas, Numpy and TensorFlow libraries.

First, a study of the data set is made, which are of time series, and all the possible information is extracted so that they can be used in machine learning models. We divided these data into three different groups in order to study the behavior of the training of the models as a function of these groups.

Next, three generative neural models are generated which will be trained with the three data groups, which gives us a total of nine different trainings, proceeding to choose after the training the models with the best performance to generate new synthetic data.

Finally, it is verified that these synthetic data have the same properties as the original data, using an autoencoder model for this, and four different trainings are carried out to compare the results between the original data, the generated data and a combination of both.

It has been decided to use three combinations between model and data group to generate the data, the synthetic data generated by the three has greatly improved the training of the autoencoders that used them.

Finally, this work has concluded that the objectives set at the beginning have been achieved. The increase of data for the training of time series data, which was the ultimate goal of the work, has obtained very good results, achieving improvements of ninety percent.

Índice

Contenido

Glosario

Pd (*Pandas*) Librería de Python que proporciona herramientas para el uso y análisis de datos.

Np (*Numpy*) Librería de Python que proporciona herramientas y recursos para trabajar con matrices de pequeño y gran tamaño.

Df (*Dataframe*) Objeto de Pandas que consiste en un conjunto de datos estructurados en dos dimensiones, donde las columnas tienen distintas propiedades.

Array Objeto de Numpy que consiste en una Matriz de valores del mismo tipo indexados por una tupla.

Machine learning Disciplina de la inteligencia artificial que se caracteriza por generar modelos que aprenden ellos solos.

Tf (*TensorFlow*): Biblioteca de código abierto que permite desarrollar y entrenar modelos de *machine learning*.

Keras: Interfaz de programación de aplicaciones de alto nivel de TensorFlow para construir y entrenar modelos de aprendizaje profundo.

AE (*Autoencoder*): Tipo de red neuronal que tiene como objetivo aprender una nueva representación de los datos con menor dimensionalidad que la entrada.

VAE (*Variational autoencoder*): Modelo de aprendizaje profundo que se caracteriza por ser un autoencoder cuya distribución de codificaciones se regulariza durante el entrenamiento con el fin de asegurar que su espacio latente tenga buenas propiedades.

MSE (*Mean squared error*): Método para medir la distancia entre el estimador y la medida estimada.

Dataset (Conjunto de datos): Conjunto de datos con unas mismas características.

NaN (*Not a Number*): Se utiliza para referirse a la representación de una indeterminación.

Layers: En español capa, es el término que designa a un conjunto de nodos que trabajan juntos en una profundidad especifica en una red neuronal.

RNN (*Recurrent neural network*): En español red neuronal recursiva, es la clase de inteligencia artificial que es capaz de almacenar información de las anteriores observaciones.

LSTM (*Long short-term memory*): Arquitectura básica de RNN.

PCA (*Principal component analysis*): Técnica usada para reducir la dimensionalidad de un conjunto de datos.

Data augmentation: Técnica que se usa para incrementar el número de datos añadiendo copias modificadas de los datos que se dispone o generando nuevos datos sintéticos.

Loss *(loss function)*: En español función de perdida, es la función que se utiliza para decidir si el entrenamiento de un modelo mejora.

Boxplot: Gráfica que permite observar la distribución de un conjunto de datos, así como los cuartiles.

Lagplot: Diagrama especial de dispersión en que las dos variables están retardadas una cantidad fija de tiempo.

Overfitting: Error de modelaje en que la función que se intenta aproximar se ajusta demasiado a un grupo concreto de datos.

Introducción y objetivos

Introducción

Debido al aumento de la capacidad computacional y la importancia que han ganado los datos en estos últimos años, el *Machine Learning* ha incrementado exponencialmente su importancia ya que está demostrando ser una herramienta muy potente y capaz en el manejo de datos, lo cual está provocando que se realicen grandes avances.

Dentro del *Machine Learning* encontramos las redes neuronales artificiales las cuales, simulando las redes neuronales biológicas, están consiguiendo ponerse a la vanguardia en muchos campos, ya sea a pequeña o a gran escala.

Las redes neuronales tienen la necesidad de ser entrenadas antes de su implementación y para ello se necesitan los datos que van a gestionar, motivo por el cual un problema que se ha podido observar actualmente es la dificultad de conseguir un buen volumen de datos para el entrenamiento de modelos, ya sea por la imposibilidad de conseguirlos actualmente o por su alto coste monetario o intelectual. En este punto es donde aparece el concepto de la *data augmentation*.

El aumento de datos es una herramienta muy poderosa para aumentar el tamaño del conjunto de entrenamiento y, por lo tanto, mejorar el proceso de aprendizaje de las redes neuronales.

Es una técnica bien estudiada en la generación de imágenes, videos o en el audio, pero no está tan desarrollada para otros tipos de datos como las series de tiempo.

Aunque existen métodos de aumento de datos de series de tiempo "clásicos", hasta donde sabemos, no existen métodos basados en redes neuronales.

Objetivos

Este trabajo tiene como objetivo principal el estudio del uso de modelos de redes neuronales para que a partir de un grupo de datos sean capaces de generar nuevos datos de serie temporal. Los datos generados deben cumplir con el requisito de que tengan las mismas propiedades y características que los datos de entrada para que otras redes neuronales puedan entrenar con mayor número de datos. Es decir, se desea conseguir una buena calidad de *data augmentation* para datos de serie temporal usando modelos de redes neuronales.

Esta meta se pretende conseguir mediante modelos de redes neuronales generativos, por lo que otro objetivo de este trabajo consiste en estudiar e implementar modelos generativos.

Por último, con este trabajo también se pretende alcanzar el objetivo de estudiar los distintos tipos de variables que tienen los datos de serie temporal y cómo estas afectan al entrenamiento de nuestra red neuronal.

Por lo tanto, se pretende con este trabajo cumplir los siguientes tres objetivos:

- Estudio de la influencia de las variables de los datos temporales.
- Estudio e implementación de modelos generativos.
- *Data augmentation* para datos de serie temporal.

Alcance del proyecto

Con el fin de que el proyecto alcance los objetivos anteriormente mencionados se pretende hacer un estudio práctico enfocado a un conjunto de datos de serie temporal concreto. Se realizarán tantos modelos de redes neuronales como se crea necesario.

Se realizará mediante el lenguaje de programación Python. Para el uso y trabajo de los datos se usan tres módulos: *datetime*, *pandas* y *numpy*. Para la creación de modelos de redes neuronales se usa Keras de TensorFlow.

El procedimiento consiste en la generación de nuevos datos sintéticos y el análisis de su capacidad para cumplir con los objetivos del proyecto. Para la generación de datos temporales se usa un modelo de tipo VAE y para analizar los resultados se utiliza *Autoencoders*.

1. Introducción a la generación de datos temporales

En este primer apartado se realiza una introducción teórica a los datos de serie temporal y a los Variational Autoencoders.

1.1 Datos de serie temporal

Los datos de serie temporal son aquellos datos que están ordenados mediante un criterio temporal, es decir, el orden de los datos proporciona una información y es clave para la interpretación de estos. Algunos ejemplos de este tipo de datos son: el número de vuelos mundiales diarios, el crecimiento de la población a lo largo de la historia, el valor de las acciones de una empresa en los últimos años…
Como se puede comprobar hay muchos datos de serie temporal. El estudio de estos permite encontrar relaciones entre las muestras como tendencias, autocorrelaciones o variaciones estacionales. En la ilustración 1 se muestra un ejemplo de este tipo de datos graficados.

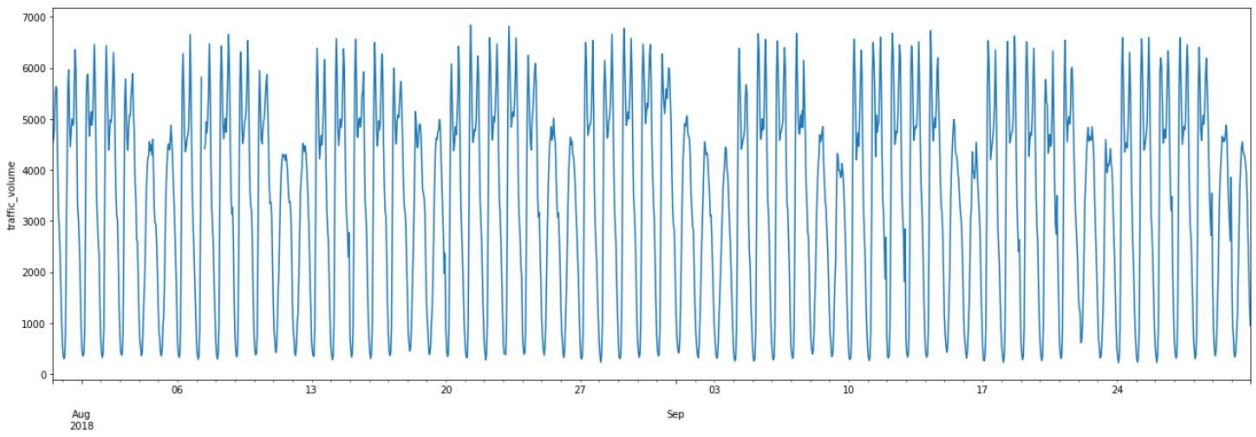

Ilustración 1 Datos en serie de tiempo

Por lo tanto, para que un conjunto de datos sea de serie temporal debe tener al menos una variable que sea de tal tipo temporal. Si en un conjunto de datos solo una variable fuera temporal, todo el conjunto de datos se convierte en serie temporal. Aunque hay que tener en cuenta que la información que aporta el carácter temporal de los datos puede decidir usarse o no, esto puede depender del objeto de estudio, es decir se puede decidir eliminar el atributo de carácter temporal para que el conjunto de datos deje de ser de serie temporal, de tal manera, que en tal caso, sería hacer que el orden de las observaciones fuera indiferente.
En nuestro caso, sí que se está interesado en utilizar la información que la serie temporal proporciona al conjunto de datos, por lo que entonces debemos introducir tres propiedades de este tipo de datos: el periodo, el paso de tiempo y la muestra.

- El periodo se define como el intervalo de tiempo que existe entre una observación y su consecutiva. Por lo general el periodo es constante a lo largo de la serie, si no fuera así se debe

tratar pues todos los modelos, como se explica más adelante, asumen que el periodo es el mismo en todo el conjunto de datos.

- Una muestra se puede referir a una observación o un conjunto de observaciones que son tratadas como grupo compacto.
- El paso de tiempo, al que también se le puede llamar tamaño de muestra, es el periodo de tiempo que hay entre la primera y la última observación que comprende una muestra. Se obtiene mejores resultados si el paso de tiempo se escoge en función de un periodo con consistencia, por ejemplo, una semana o un mes. El paso de tiempo debe ser lo suficientemente grande para que contenga información, pero no demasiado grande para disponer de suficientes muestras para el entrenamiento de las redes neuronales.

Una vez se han visto estas características de los datos de serie temporal, se puede pasar a ver los distintos tipos de variables y su interpretación para su uso. Existe tres tipos de variables: las variables categóricas, las variables discretas y las variables continuas.

- Las variables continuas son aquellas que pueden tomar un número infinito de valores entre dos valores. En los estudios que involucran datos de serie temporal, generalmente las variables continuas son el objetivo del estudio, ya sea porque se desea predecir un valor, monitorizar el comportamiento, estudiar patrones o generar nuevos datos.
- Las variables discretas son variables numéricas que solo pueden tomar valores reales entre dos valores. Estas variables tienen un carácter parecido a las variables continuas y cuando se trabaja con datos temporales no se diferencian mucho entre ellas.
- Finalmente, las variables categóricas son aquellas que contienen un número de categorías o grupos finitos. Este tipo de variables pueden tener valor numérico o literal. Para el estudio de datos con modelos de redes neuronales se debe trabajar solo con valores numéricos, por lo tanto, si se tiene un conjunto de datos con variables categóricas con literales se deben transformar a numéricas.

Toda variable continua puede ser transformada a discreta y a categoría, las variables discretas también se pueden transformar a categóricas. Aunque se puedan realizar dichos cambios, hay que tener en cuenta que en toda transformación se pierde algo de información. Estos cambios se pueden interpretar como una sintetización y en algunos casos pueden ser útiles para que el modelo reciba información más concreta y concisa.

Visto esto, ya se ha sintetizados todos los conceptos que se deben tener en cuenta sobre los datos de serie temporal antes de empezar a trabajar con redes neuronales.

1.2 Redes neuronales recursivas

Una vez se ha visto las características principales de los datos de serie temporal se puede pasar a ver las redes neuronales recursivas, que son una clase de red neuronal artificial que se caracteriza por tres rasgos:

1. Capacidad de tratar elemento a elemento independientemente de la secuencia de entrada.
2. Capacidad de recordar las salidas anteriores y tomar decisiones en función de éstas.
3. Capacidad de tratar datos secuenciales de forma eficiente.

Las tres características, -especialmente las dos últimas-, las hace una clase de red neuronal muy interesante para el tratamiento de los datos temporales, pues encajan muy bien con las propiedades de este tipo de datos, ya que, los datos de serie temporal se caracterizan porque el orden en que se presentan tiene sentido y por lo tanto contiene información.

La contrapartida de este tipo de clase de red neuronal es el aumento del coste de procesamiento, con todas las consecuencias que puede generar: altos tiempos de entrenamiento, desvanecimiento de gradiente, etc., debido al aumento de la capacidad.

Este concepto se ve implementado con la red *Long short-term memory*. Una unidad básica de LSTM está formada por una puerta de entrada, una puerta de salida, una célula de memoria y una puerta de olvido. La estructura general de este tipo de configuración se muestra en la Ilustración 2.

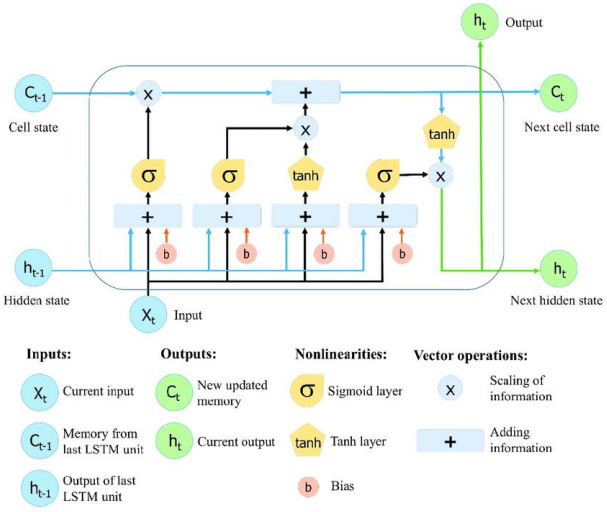

Ilustración 2 Lee, Giha. Application of Long Short-Term Memory (LSTM) Neural Network for Flood Forecasting. [Ilustración]. Recuperado de https://www.researchgate.net/figure/The-structure-of-the-Long-Short-Term-Memory-LSTM-neural-network-Reproduced-from-Yan_fig8_334268507.

Gracias a esta ilustración se puede ver como se sintetizan todos los conceptos que se ha visto de las RNN. La LSTM tiene tres puertas de entrada y puede tener dos o tres puertas de salida. Se puede ver también que una unidad de LSTM está formada por cuatro *layers*.

1.3 Variational Autoencoders

Vistos los conceptos relacionados con los datos de serie temporal y la clase de RNN, ahora se pasa a estudiar cómo generar datos con redes neuronales.

Existen dos tipos de modelos de redes neuronales con los que se puede obtener buenos resultados para la generación de nuevos datos: Los Generative adversial network y los Variational Autoencoders.

En este trabajo se usa el Variational Autoencoder, para cuyo entendimiento primero se debe comprender que es un Autoencoder.

Autoencoder

El Autoencoder es un tipo de red neuronal con la particularidad que la dimensión de la entrada es la misma que la dimensión de la salida. A su vez todos los *hidden layers* tiene una dimensionalidad más reducida que en la entrada, siendo la capa oculta central la de dimensión mínima. Esto hace que el Autoencoder tenga una estructura en forma de pajarita, como se muestra en la Ilustración 3.

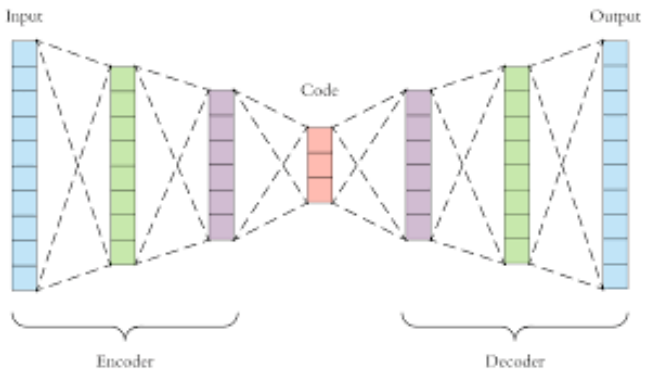

Ilustración 3 Dertat, Arden. (2017). Aplied Deep Learning – Part 3: Autoencoders. Recuperado de https://towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798

El autoencoder por lo tanto forma un cuello de botella en su capa oculta central, al que se le llama espacio latente o *latent space* en inglés. El espacio latente divide el Autoencoder en dos partes: el encoder y el decoder.

- El encoder es la estructura que se encuentra antes del espacio latente y se encarga de reducir la dimensión de los datos de entrada.

- El decoder es la estructura que se encuentra después del espacio latente y se encarga de devolver la dimensión original a los datos de entrada.

El objetivo del autoencoder es aprender una nueva representación de los datos con la dimensión del espacio latente, intentando perder el mínimo de información. Por lo tanto, el autoencoder reduce la dimensionalidad de los datos de entrada para volver a reconstruirlos. Por ello se entrena al modelo para reducir el error de reconstrucción. Se puede comparar la función del autoencoder con la función del PCA.

Por lo tanto, cada punto del espacio latente corresponde a una muestra de los datos. El problema que encuentra el Autoencoder para la generación de datos cuando se encuentra en este punto es que no tiene el espacio latente ordenado y por lo tanto no es capaz de generar datos con sentido. Este problema lo solucionan los modelos tipo VAE que se explican a continuación.

Variational Autoencoder

Los VAE han obtenido buenos resultados en la generación de datos como por ejemplo en la reconstrucción de caras, imágenes, caracteres escritos a mano… Los Variational Autoencoders también pueden ser de la clase de red neuronal recursiva y es por esto por lo que también son usados para tratar datos de serie temporal con buenos resultados. Pero para ser capaces de generar nuevos datos deben solucionar los problemas que tienen los AE en este propósito.

Se puede definir un VAE como un tipo de red neuronal con la misma estructura y características que el Autoencoder, pero donde se le ha modificado el espacio latente y se le ha cambiado el objetivo en su entrenamiento con el fin de que el espacio latente quede ordenado.

El VAE respecto al AE interpreta de forma distinta el espacio latente, ya que éste debe ser continuo y debe estar bien distribuido para poder escoger un punto con el que se pueda obtener un nuevo valor, es decir, generar nuevos datos. Con este objetivo el autoencoder interpreta el espacio latente como un vector de medias y un vector de desviaciones estándar, que unidos adecuadamente forman un vector de variables aleatorias Normales. En la Ilustración 4 se muestra la estructura de un modelo tipo VAE donde se puede ver también la estructura de su espacio latente. Esta configuración permite tener el espacio latente ordenado.

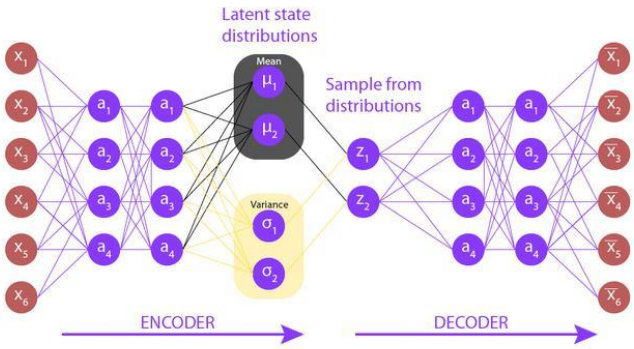

Ilustración 4 pawangfg. (2020). Variational Autoencoders. Recuperado de https://www.geeksforgeeks.org/variational-autoencoders/

Aunque el VAE tenga la posibilidad de tener el espacio latente ordenado no tiene porque ordenarlo en el entrenamiento. Para conseguir que el modelo ordene el espacio latente lo que se hace es dividir la función de pérdida en dos términos: el término de la reconstrucción de los datos y el término que se encarga de ordenar el espacio latente.

2. Los datos de trabajo

Hecha la introducción teórica ahora se pasa a explicar las características del conjunto de datos que se usa para realizar este trabajo.

2.1 El dataset

El dataset que se utiliza en este trabajo está titulado *Metro Interstate Traffic Volume*, obtenido de la página web de recursos abiertos "*UCI Machine learning repository"*, que ofrece conjuntos de datos especialmente seleccionados para trabajos de *machine learning*.

El conjunto de datos representa el volumen horario de tráfico de un metro entre dos estaciones cerca de Minneapolis, Estados Unidos. En la Tabla 1 se muestra los atributos del dataset.

Attribute	Type	Descripción
Holiday	Categorical	Fistas nacionales y regionales
temp	Numeric	Temperatura en Kelvin
rain_1h	Numeric	Precipitación en mm
snow_1h	Numeric	Cantidad de nieve en mm
clouds_all	Numeric	Porcentage cubierto de nuves
weather_main	Categorical	Descripción corta del tiempo
weather_description	Categorical	Descripción larga del tiempo
date_time	Date Time	Hora de la fecha en CST
traffic_volume	Numerical	Información del volumne del trafico

Tabla 1 Atributos del dataset

De esta información se extrae que los datos objetivo del estudio se encuentran en el atributo *traffic_volume*. Se ve también que *date_time* es el atributo temporal y que proporciona la fecha y la hora en que fue tomada la muestra, por lo tanto, se deduce que el periodo de este *dataset* es una hora.

En la Ilustración 5 se muestra la evolución de las últimas muestras del volumen de tráfico en función del tiempo en que fueron tomadas, pudiéndose observar que hay un patrón que se repite cada cierto tiempo, una semana, con dos días en que el volumen de tráfico disminuye considerablemente, siendo lógico pensar que se trata de los fines de semana. También se observa que el comportamiento a lo largo de un día es constante a lo largo de las semanas con dos picos de volumen de tráfico a principio y a final de día.

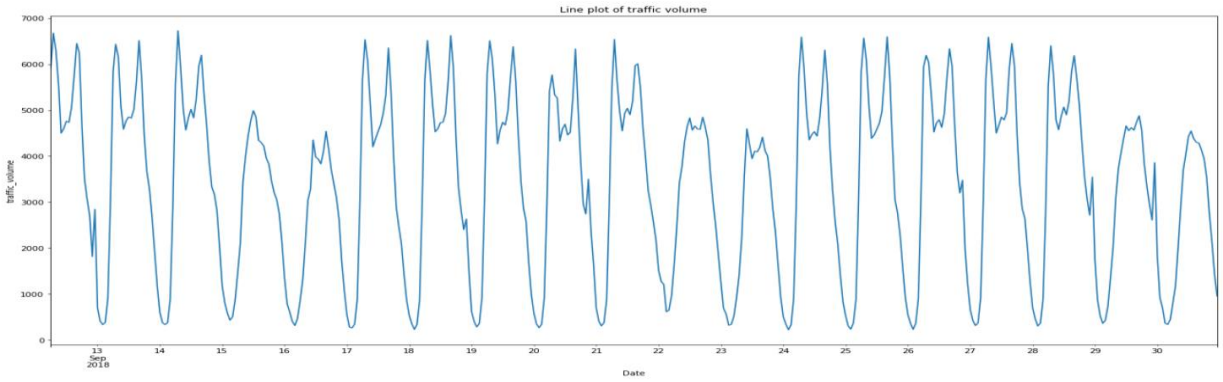

Ilustración 5 Evolución temporal del volumen de tráfico

El conjunto de los datos también contiene cuatro variables numéricas (*temp, rain_1h, snow_1h, clouds_all*) y tres categóricas (*holiday, weather_main, weather_description*) que proporcionan una información adicional para cada observación que se puede usar para mejorar el objetivo del trabajo.

El conjunto de datos que se obtiene tiene 40.575 muestras, teniendo todas ellas todos los atributos conocidos, esto es, sin que valores NaN. Si se rellenan las horas que faltan en la serie temporal se obtiene 52.551 muestras. Por lo tanto, se tiene 12.000 muestras de las cuales no se dispone y por lo tanto pasan a ser representadas con un valor NaN en todos sus atributos.

A continuación, se representan las muestras que faltan a lo largo del tiempo para ver cómo se distribuyen. Se obtiene el resultado que se muestra en la Ilustración 6, representándose en blanco las fechas en que faltan datos y en negro las fechas de las que se tienen datos.

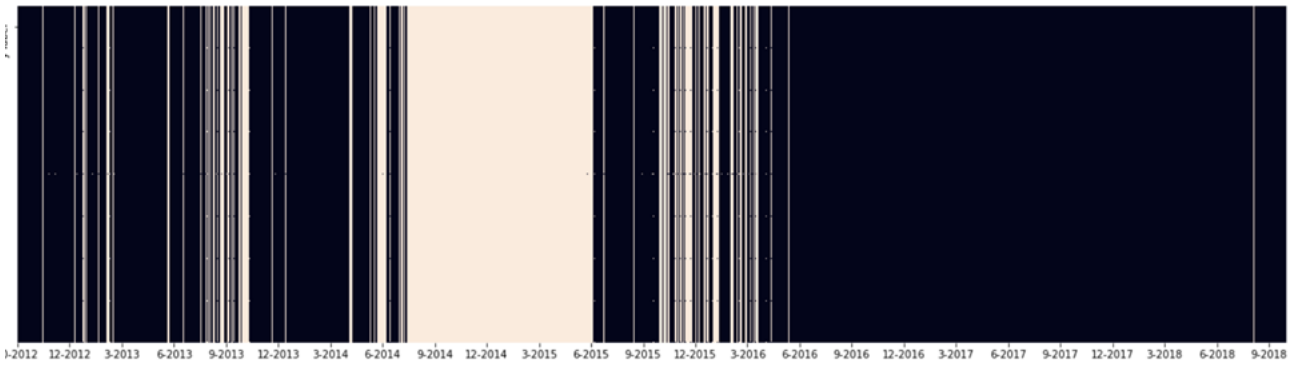

Ilustración 6 Heat Map con las muestras que faltan

Se ha observado que hay mucha falta de muestras entre junio de 2014 y junio de 2015. También se encuentra una serie de muestras de las que no se tiene valor entre 2012 y medianos de 2014 y entre medianos de 2015 y medianos de 2016, pero esta vez en intervalos más pequeños. Por otra parte, desde medianos de 2016 hasta el final de la serie se ha podido observar la ausencia de muestras es ocasional o puntual.

No es recomendable usar datos con muchos valores NaN consecutivos, pues estos se deben rellenar con algún método y esto genera un ruido en los datos, haciendo que la información que nos interesa quede enmascarada por información que no es real.

Debido a lo expuesto se decide trabajar solo con los datos a partir del día 1 de mayo de 2016. En este periodo se encuentran muy pocas faltas de muestras, a pesar de lo cual deben ser rellenadas, a cuyo efecto se usa el método de la interpolación para los atributos de tipo numérico, y para los atributos categóricos se rellenan con la última observación hecha del mismo atributo.

En la Ilustración 7 se muestra el *DataFrame* resultante, habiéndose asignado la serie temporal como índice para poder usar mejor el conjunto de datos, obteniéndose un resultado con 21.192 muestras, las cuales se han reducido a la mitad respecto al conjunto de datos iniciales, pero son muestras suficientes para el entrenamiento y ya están formadas solo por valores conocidos.

	holiday	temp	rain_1h	snow_1h	clouds_all	weather_main	weather_description	traffic_volume
2016-05-01 00:00:00	None	280.75	0.0	0.0	90.0	Rain	light rain	1628.0
2016-05-01 01:00:00	None	280.02	0.0	0.0	90.0	Clouds	overcast clouds	946.0
2016-05-01 02:00:00	None	279.48	0.0	0.0	90.0	Clouds	overcast clouds	735.0
2016-05-01 03:00:00	None	279.02	0.0	0.0	90.0	Clouds	overcast clouds	395.0
2016-05-01 04:00:00	None	278.56	0.0	0.0	90.0	Drizzle	drizzle	390.0

Ilustración 7 Encabezado del DataFrame despues de tratar los NaN

Una vez hemos eliminado los valores NaN, ya disponemos de todos los datos finales, pero antes de poder entrenar los modelos con ellos los debemos someter a los procesos que explicamos en los dos siguientes subapartados.

2.2 Categorización y análisis gráfico de los datos

Categorización

Antes de separar los datos para el entrenamiento, decidimos realizar una transformación de todas las variables que no son el objetivo de estudio, que recordamos que es *traffic_volume* a categóricas. Esto lo hacemos con el fin de sintetizar la información para el entrenamiento del modelo, hecho que mejora su velocidad de entrenamiento y que en muchos casos puede mejorar el resultado.

Para hacer la categorización separamos las variables en tres grupos, decididos por el carácter de la variable original.

Al primer grupo se lo llama variables temporales y está formado por las variables que se refieren al tiempo, y que automáticamente son categóricas, mostrándose su resumen en la tabla 2.

Variable	Descripción	Tipo	Rango
Month	Mes en que fue tomada la muestra	Categórica	1-12
Day	Día de la semana en que fue tomada la muestra	Categórica	0-6
Hour	Hora en que fue tomada la muestra	Categórica	0-23

Tabla 2 Variables temporales

Al segundo grupo se le llama variables categóricas descriptivas, las cuales están formadas por las variables categóricas que están representadas con un valor no numérico y se deben factorizar para que puedan ser interpretadas por la red neuronal, mostrándose las mismas, así como sus propiedades después de la factorización realizada, en la tabla 3.

Variable	Descripción	Tipo	Rango
holiday	Si el día de la muestra es festivo o no	Categórica	1-2
weather_main	Descripción corta del tiempo	Categórica	0-9
weather_description	Descripción larga del tiempo	Categórica	0-33

Tabla 2 Variables categóricas descriptivas

Finalmente, el tercer grupo son las variables numéricas, que son las variables numéricas continuas las cuales se deben transformar a variables categóricas y se sintetizan en la Tabla 3.

Variable	Descripción	Tipo	Rango
temp	Si el día de la muestra es festivo	Categórica	1-5
rain_1h	Descripción corta del tiempo	Categórica	1-4
snow_1h	Descripción larga del tiempo	Categórica	0

Tabla 3 Variables numéricas

Este último grupo tiene algunas propiedades que se deben comentar. Para la variable *temp* se ha decidido dividir en cinco percentiles y clasificar las muestras en función del que pertenezcan. Para la variable *rain_1h* se ha decidido dividir en cuatro teniendo en cuenta la media. Para la variable *snow_1h* se ha visto que no ha nevado ningún día y por lo tanto tiene siempre el mismo valor, no teniendo sentido introducir esta variable en el estudio que se está haciendo, por lo que se decide descartarla.

Análisis grafico de los datos

Una vez hecha la categorización de las variables auxiliares se pasa a hacer un análisis visual de su afectación en el volumen de tráfico, para ver gráficamente qué variables pueden influir más en el parámetro objetivo. En los *boxplots* de la Ilustración 6 se muestra los resultados obtenidos.

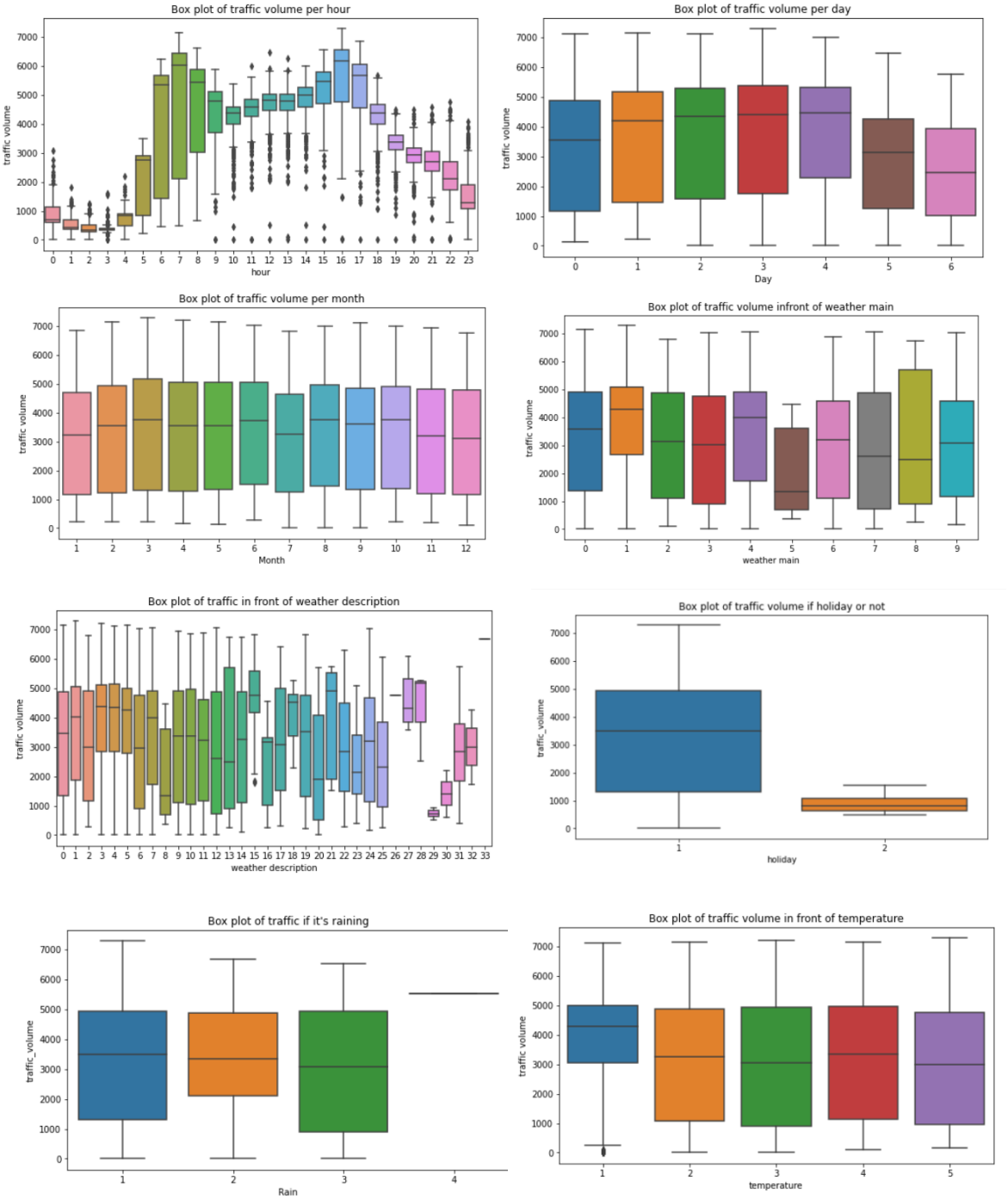

Ilustración 6 Box plots para cada variable categórica enfrente el volumen de trafico

Se puede observar que hay variables que afectan mucho al resultado como es el caso de *Hour*, *Day* o *Holiday*. Las dos primeras ya se habían observado anteriormente con la gráfica de línea que se había

hecho de los datos al principio de todo, y sobre la variable *holiday* es lógico pensar que disminuye mucho el volumen de tráfico en los días de fiesta.

Por otra parte, es lógico pensar que al hacer peor tiempo, el volumen de tráfico debería aumentar, pero se puede observar, con la ayuda de las variables *temp, weather_description, rain* y *weather_main*, que esta tendencia, aunque sí que se note no es tan marcada como se esperaría.

Por lo tanto, los grupos de variables que se ha visto que son más importantes en orden descendente son las siguientes: variables temporales, variables categóricas descriptivas y variables numéricas. Esto se tendrá en cuenta al realizar el modelo.

Seguidamente se pasa a estudiar la correlación que tienen las muestras con las muestras hechas anteriormente, para lo cual se ha realizado el *lag plot* que se muestra en la Ilustración 7. El hecho de que los puntos se acumulen de abajo a la izquierda a arriba a la derecha sugiere que existe una correlación positiva, aunque haya algunos dispersos que pueden haber sido provocados por algún suceso poco común que haga variar mucho el volumen de tráfico. Se puede creer que esto es debido a la variable *holiday*, pero después de una comprobación se ha visto que no es así. En todo caso parece una variable interesante de tratar con redes neuronales recursivas porque se observa una buena correlación entre la sucesión de muestras.

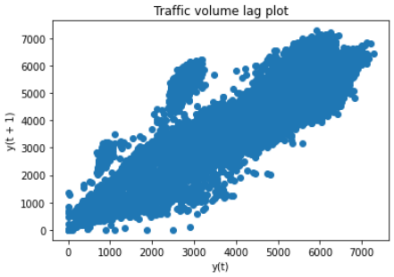

Ilustración 7 Lag plot del volumen de tráfico

Finalmente se decide hacer un histograma para ver que distribución siguen las muestras sin tener en cuenta el carácter temporal de ellas, el cual se representa en la Ilustración 8.

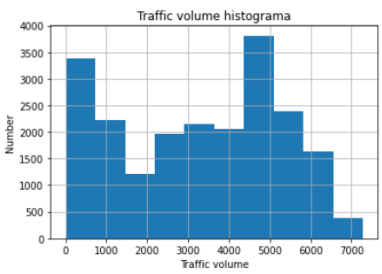

Ilustración 8 Histograma del volumen de tráfico

Parece que para valores altos de volumen de tráfico las muestras siguen una distribución gaussiana, aunque si se tiene en cuenta los valores bajos, entre 0 y 2000, no parece apreciarse ningún tipo de distribución conocida. Por lo tanto, no se ha podido extraer información de este tipo de gráfico.

2.3 División de los datos y escalado

Llegados a este punto se pasa a hacer la división de los datos, siendo éste el último paso que se debe dar con los datos para su uso en el entrenamiento de redes neuronales.

Primeramente, se decide escoger un paso de tiempo de 168, correspondiente a una semana (veinticuatro horas por siete días), que como se ha visto en la primera exploración de la variable objetivo *traffic_volume*, es un paso de tiempo que tiene suficiente información para que contenga información relevante. Se decide también solapar las muestras con el objetivo de conseguir muchos más datos de entrenamiento, hecho que permite entrenar mejor el modelo.

Se han separado los datos en dos grupos: entradas categóricas y la entrada objetivo (*traffic_volume*) ya que los datos deben seguir caminos separados en el entrenamiento de la red debido a su distinto objetivo. Por lo tanto, la malla de datos que resulta de estos dos procesos es la que se muestra en la Ilustración 9. Aunque el número de variables categóricas sea seis en este caso va a depender del modelo, lo cual explicaremos posteriormente.

```
((21024, 168, 6), (21024, 168, 1))
```

Ilustración 9 Malla de los datos categóricos y de los datos objetivos

Seguidamente se realiza la división en datos de entrenamiento y datos de test, para lo cual decidimos usar una proporción de 80-20, que obtiene muy buenos resultados en general. Realizado este paso el tamaño de los datos resulta en la malla que se muestra en la Ilustración 10.

```
(16819, 168, 6) (4205, 168, 6)
(16819, 168, 1) (4205, 168, 1)
```

Ilustración 10 Malla de los datos de entrenamiento y de test

Antes de poder pasar los datos al modelo de red neuronal se deben escalar, hecho que facilita mucho el entrenamiento. Para ello se decide normalizar los valores entre -1 y 1, habiéndose optado por este rango porque en términos generales se obtiene un mejor rendimiento que utilizando el rango entre 0 y 1 o cualquier otro.

En este punto ya disponemos de los datos preparados para ser usados como entrenamiento para la red neuronal.

3. Procedimiento de trabajo

El procedimiento general del trabajo es el que se muestra en la Ilustración 11.

Ilustración 11 Procedimiento del trabajo

Primeramente, se entrena los modelos generativos tipo VAE, para a continuación, a partir de los modelos entrenados, proceder a generar nuevos datos, y así, finalmente poder proceder a analizarlos mediante AE.

Este proceso se quiere repetir por tres modelos generativos distintos y para cada modelo con tres grupos de datos distintos, todo lo cual deja un total de 9 ejemplos distintos para poder comparar resultados.

Los distintos modelos generativos se explican en el subapartado siguiente (3.1 Modelos generativos), en cuanto a los tres grupos de datos va referido a los datos categóricos de las entradas no objetivo, los cuales se decide hacer la separación en grupos siguiente:

1. Variables categóricas temporales.
2. Variables categóricas temporales + Variables categóricas descriptivas.
3. Variables categóricas temporales + Variables categóricas descriptivas + Variables categóricas numéricas.

Como se puede comprobar, las variables categóricas temporales se encuentran en los tres grupos, esto se debe a dos razonamientos:

- En la exploración de los datos, se ha comprobado que estas variables son las que más influyen en la variable objetivo, por lo que, si se prescindieran de ellas, se perdería mucha información.
- Las variables categóricas temporales están en todos los datos de serie temporal, recordando en este sentido que, para que un conjunto de datos sea de serie temporal, tiene que concurrir al menos una característica temporal. Por consiguiente, estas variables se encuentran en cualquier otro problema que se realice con este tipo de datos.

En el segundo grupo se decide añadir las variables categóricas descriptivas a las temporales ya que se ha visto que estas variables están bastante correlacionadas con la variable objetivo, no tanto como las variables temporales, pero se cree que puede mejorar notablemente el resultado.

El último grupo está formado por todas las variables que nos proporciona el conjunto de datos, menos la *snow_1h* que en la porción de datos que se ha seleccionado del total se ha podido comprobar que era constante. Este grupo es el que proporciona más información al modelo y por lo tanto el que en teoría debe obtener mejores resultados, pero debido a que las variables categóricas numéricas se ha encontrado que están poco correlacionadas con la variable objetivo se espera que la mejora respecto al anterior grupo de variables sea mínima.

Teniendo claro cómo funciona el procedimiento general de trabajo ahora se pasa a explicar cómo funciona cada elemento de este procedimiento.

3.1 Modelos generativos

Estructura de trabajo de los modelos generativos

El procedimiento de trabajo para el modelo generativo sigue la estructura que se explica a continuación y se muestra en la Ilustración 12.

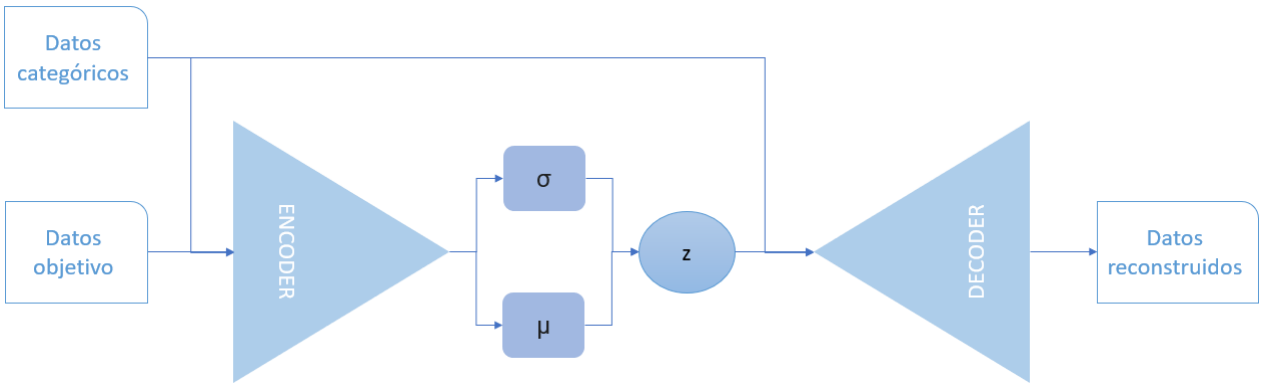

Ilustración 12 Estructura de trabajo del modelo

En primer lugar, se introduce la secuencia de datos objetivo y la secuencia de datos categóricos en el encoder, el cual al estar formado por layers LSTM recibe en el input la secuencia de tres dimensiones y genera un output en que la secuencia tiene dos dimensiones. El output del encoder se divide para formar la media y la desviación estándar del espacio latente, a los cuales se le aplica la ecuación 1 para poder generar la distribución gaussiana del espacio latente.

$$z = \mu + \exp(0{,}5 * \sigma) * \varepsilon$$

Ecuación 1 Generación del espacio latente

Donde μ es el vector de medias, σ es el vector de desviaciones estándar que proporciona la salida del encoder, ε es una normal de media cero y desviación estándar uno y la z representa el espacio latente.

A continuación, el decoder recibe en la entrada la salida del espacio latente el cual transforma de dos a tres dimensiones, y tras la referida transformación, ésta se concatena con los mismos datos categóricos de entrada en el encoder. La salida del decoder es la muestra reconstruida, por lo que tiene la misma dimensión que los datos objetivo de entrada al encoder.

En el entrenamiento se debe imponer al modelo de VAE que minimice el error de reconstrucción y que organice correctamente el espacio latente, haciéndose de tal manera mediante la *loss function* en la que se suman los dos términos con el mismo peso. En el conjunto de ecuaciones 2 se muestra la ecuación de la función de pérdida.

Para el error de reconstrucción se ha usado la formula del MSE.

$$rec = \frac{1}{pasos\ de\ tiempo} * \sum_{i=1}^{pasos\ de\ tiempo} (x_i - y_i)^2$$

$$k1 = -\frac{1}{2} * (1 + \sigma - \mu^2 - e^{\sigma})$$

$$vae\ loss = rec + k1$$

Ecuación 2 Loss del VAE

Donde *x* es el valor de cada muestra a la entrada e *y* es el valor de cada muestra a la salida. El término que se refiere a la reconstrucción es el *rec* y el término que se refiere al orden del espacio latente es el *k1*.

Estructura interna de los modelos

Este es el procedimiento general de trabajo que se ha realizado, pero para realizarlo se han utilizado tres modelos distintos en cuanto a su estructura interna para poder analizar las diferencias en los resultados entre ellos y así poder ver cómo afecta cada cambio al resultado. Esto se hace con el objetivo de poder escoger el mejor modelo para alcanzar con mejor precisión el objetivo de este trabajo.

Lo único que cambia entre los modelos son el número de unidades de LSTM que forman los *hidden layers* del encoder y del decoder, mostrando en la tabla 5 las distintas unidades de LSTM que forman cada *layer* para cada modelo. En esta tabla, los números que van acompañados con una d representan *layers* formados por unidades Dense y los que no están acompañados por ninguna letra representan *layers* LSTM.

	Encoder	Decoder
Modelo 1	64-32d	64
Modelo 2	80-32d	80
Modelo 3	64-32-32d	32-64

Tabla 4 Estructura de los modelos

Se ha puesto un Dense hidden layer antes del espacio latente, lo cual estabiliza el resultado.

El encoder tiene una entrada referida a los datos objetivo que va directamente al *layer concatenate*, y una por cada variable categórica. Antes de pasar las variables categóricas por el *layer concatenate* se las pasa por un *layer embedding* que las transforma a un vector, permitiendo de esta manera que el modelo entienda mejor estas entradas. En la Ilustración 13 se muestra un ejemplo de cada tipo de entrada del encoder.

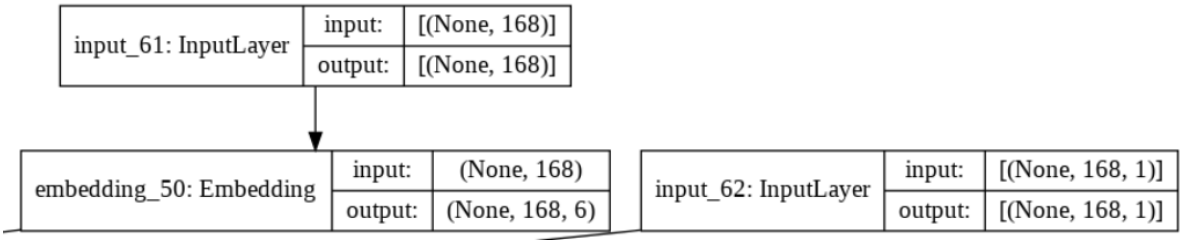

Ilustración 13 Entradas encoder del VAE

Una vez se han introducido todas las variables, se procede a introducirlas en el *layer concatenate* para poder pasar por el núcleo del encoder, es decir por los layers LSTM y los layers Dense, y que así a continuación se dividan en la media del espacio latente y en la varianza del espacio latente. Finalmente, estas dos partes se juntan para formar el espacio latente y generar la salida del encoder, mostrando este proceso en la Ilustración 14.

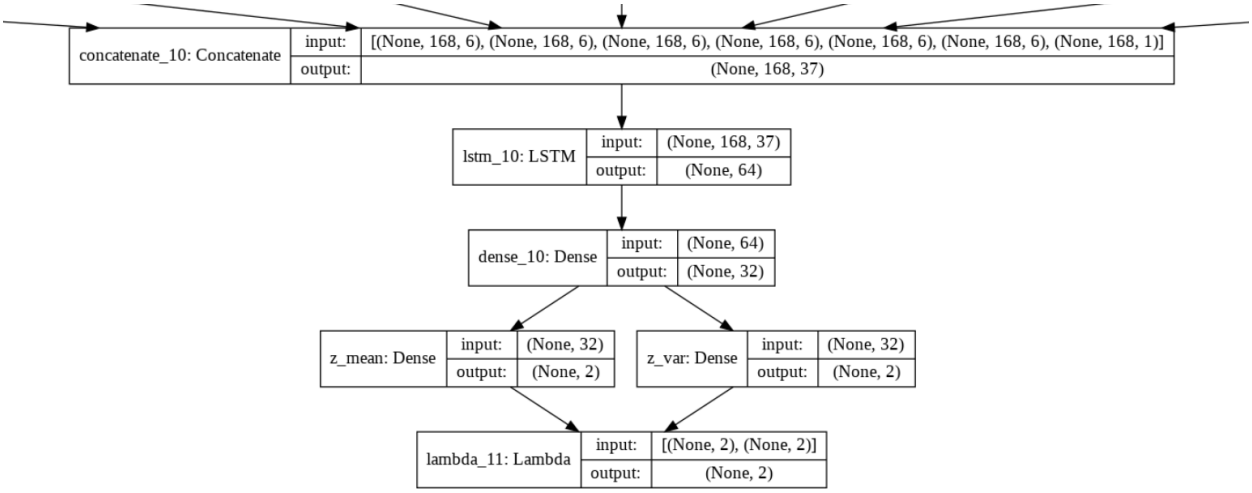

Ilustración 14 Cuerpo del encoder del VAE

El decoder es muy parecido en estructura al encoder: las entradas categóricas del encoder y el decoder son las mismas y siguen el mismo proceso. No obstante, a diferencia del encoder, en el decoder en vez de tener como entrada los datos originales tiene la representación del espacio latente. En la Ilustración 15 se muestra estas entradas.

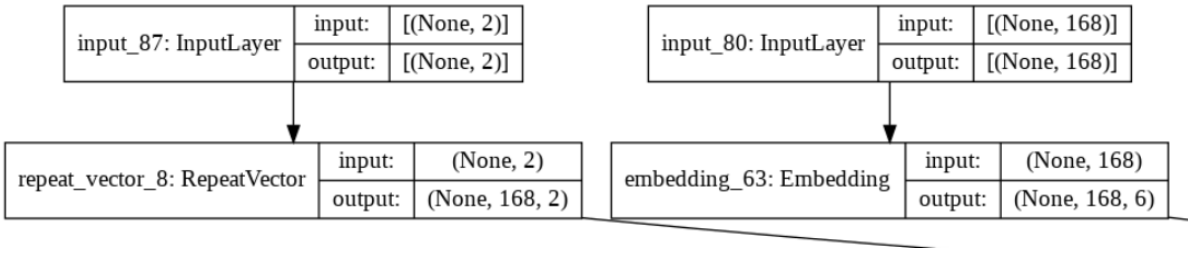

Ilustración 15 Entradas decoder del VAE

El cuerpo del decoder (Ilustración 16), también es muy parecido al cuerpo del encoder. Primero se deben concatenar todas las entradas para que los datos puedan pasar por los LSTM layers del modelo. Finalmente se debe hacer la distribución de los datos de forma que devuelva los datos con el número de características que se desee, en nuestro caso solo tenemos una característica objetivo.

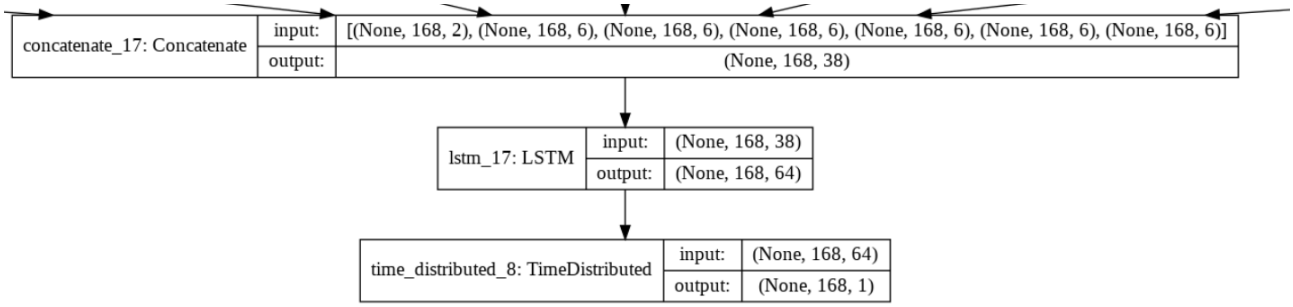

Ilustración 16 Cuerpo del decoder del VAE

Esta es la estructura del modelo generativo que se usa en el trabajo.

3.2 Generación de datos

La generación de datos se hace una vez se tenga el modelo de VAE entrenado y solo se usa el decoder del modelo generativo, necesitándose también los datos categóricos de entrada. La estructura que sigue el procedimiento de generación de datos es el que se muestra en la Ilustración 17.

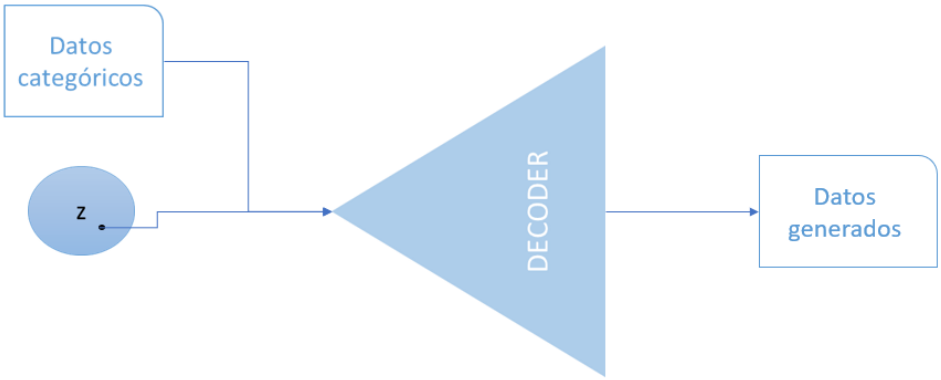

Ilustración 17 Procedimiento de trabajo para la generación de datos

Primeramente, se escoge un punto aleatorio del espacio latente que se sabe, gracias al entrenamiento del VAE, que sigue una distribución gaussiana con media de cero y desviación estándar de 1. En la Ilustración 18 se muestra un avance del resultado que se obtiene de la distribución del espacio latente.

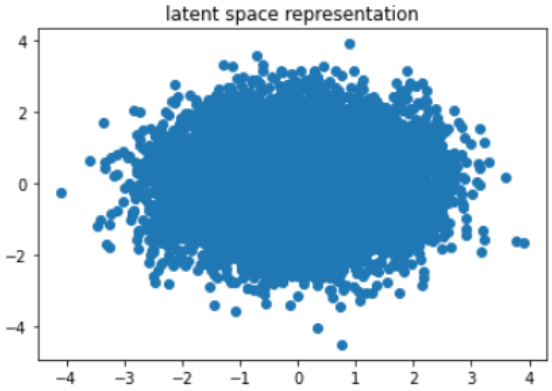

Ilustración 18 Representación del espacio latente

Se ha escogido un espacio latente de dos dimensiones y se ha visto que la distribución en el espacio no depende de la variable categórica. Por lo tanto, la muestra será más probable cuando más cerca este del punto (0,0) y viceversa. En la Ilustración 19, mostramos la distribución de las dos dimensiones gracias a la cual, efectivamente, podemos ver mejor que forma una distribución gaussiana, debiendo hacer constar que esta distribución se ha hecho dividiendo el espacio latente en 50 intervalos entre el máximo y el mínimo, como se ha comentado, siendo esto un avance necesario de los resultados para entender la generación de datos.

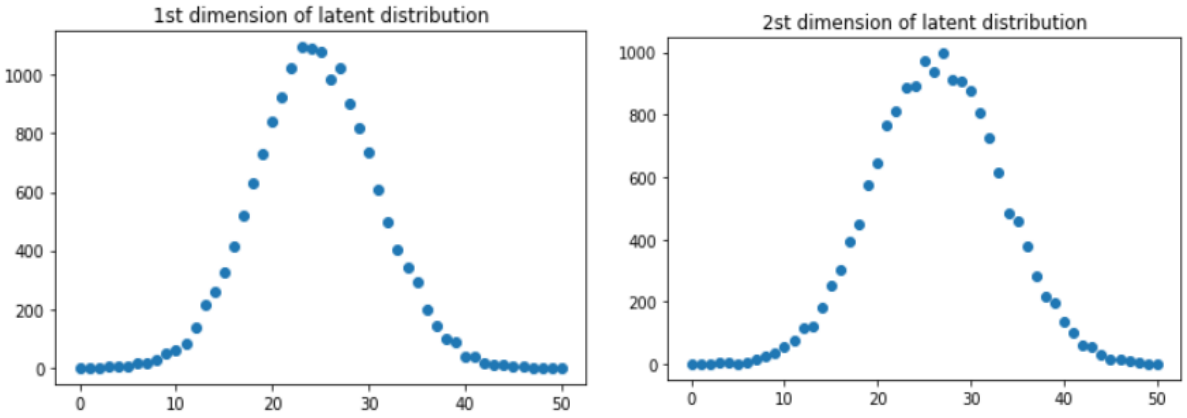

Ilustración 19 Distribución de las dimensiones del espacio latente

Una vez se ha escogido un punto aleatorio del espacio latente, mediante una distribución normal con media cero y desviación 1, se introduce en el input del decoder junto con las variables categóricas de entrada. Se decide usar las mismas variables categóricas que las de los datos originales por dos motivos:

- Escoger las variables categóricas originales permite comparar en la salida si los datos generados son distintos a los de entrada y por lo tanto nos permite comprobar que estamos generando datos totalmente nuevos.
- Escoger las variables categóricas de entrada originales permite saber que estas entradas tienen una cohesión a lo largo del tiempo y por lo tanto tienen sentido, ya que éstas son variables reales. Si se escogiera siguiendo un criterio se podría correr el riesgo de caer en situaciones sin sentido que generaría datos que no interesarían.

Al introducir los dos tipos de entradas al encoder, este ya será capaz de descodificar la información y devolver una muestra generada totalmente nueva.

3.3 Comprobación de la calidad de los resultados

En este punto ya se dispone de los datos generados, datos sintéticos totalmente nuevos en teoría con las mismas propiedades que los datos de entrada. Es en este punto donde se comprueba esta teoría. Para hacerlo se decide utilizar modelos de *Autoencoders* por tres motivos:

- Como se ha explicado anteriormente, son un tipo de red neuronal que permite comprobar los objetivos que se ha marcado este trabajo de data augmentation.
- Son modelos muy fáciles de entrenar y usan las mimas entradas y salidas que los Variational Autoencoders.
- Estudian el comportamiento de la muestra y la descomposición de sus observaciones.

El problema de los Autoencoders es que son de la misma naturaleza que los Variational Autoencoders, lo cual podría generar interferencias en el resultado, decidiéndose para evitarlo modelar cuatro modelos de Autoencoder con la misma estructura, pero entrenarlos de formas distintas. Usar esta división de entrenamientos a la vez permite analizar por comparación los resultados.

La estructura para el Autoencoder que se decide utilizar es la que se muestra en la Ilustración 20. El AE tiene solo una capa oculta de tipo Dense de 32 unidades y un espacio latente de dos dimensiones.

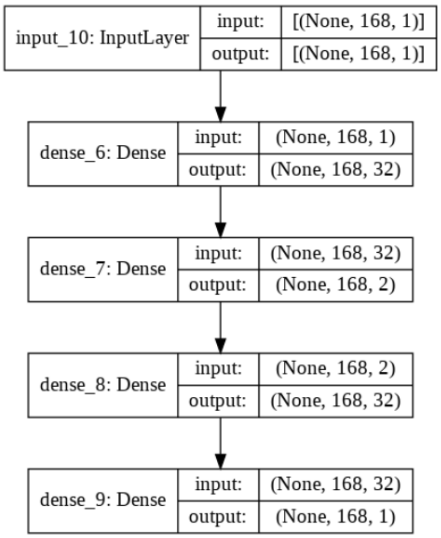

Ilustración 20 Estructura del Autoencoder

La diferencia entre las cuatro formas de entrenamiento que se usan se encuentra en los datos utilizados para entrenar cada modelo, mostrándose en la tabla 5 un resumen de éstos.

Modelo de autoencoder	**Datos que se usa para el entrenamiento**	**Datos que se usa para las pruebas**
Modelo 1	Entrenamiento VAE	Entrenamiento VAE Test VAE Generados
Modelo 2	Generados	Entrenamiento VAE Test VAE Generados
Modelo 3	0,5 * Entrenamiento VAE 0,5 * Generados	Entrenamiento VAE Test VAE Generados

Modelo 4	Entrenamiento VAE Generados	Entrenamiento VAE Test VAE Generados

Tabla 5 Datos de entrenamiento para los Autoencoders

La notación que se usa en la tabla es la siguiente: *Entrenamiento VAE* y *Test VAE* se refiere a los datos originales que se han usado respectivamente para entrenar y hacer el test del Variational Autoencoder. *Generados* se refiere a los datos que generados mediante el *Variational Autoencoder*. El 0,5 se refiere a la mitad de los datos a los que acompaña.

Para analizar los resultados de los distintos modelos de AE se decide monitorizar los valores de la *loss function* para los datos de Entrenamiento VAE, Test VAE y Generados. También se anota el tiempo de entrenamiento de los modelos.

Se decide que el primer modelo realice el entrenamiento con los datos originales de entrenamiento del VAE, para así poder comprobar que tal aproxima los valores generados en frente de los de test.

En el segundo entrenamiento el modelo se entrena con los datos que se han generado con el objetivo de poder comprobar si solo con los nuevos datos puede entender el comportamiento de los datos originales.

El tercer modelo usa de entrenamiento la mitad de los datos originales de entrenamiento del VAE y la otra mitad de los datos generados, este modelo nos sirve para ver si se compenetran bien los dos tipos de datos.

Finalmente, en el último modelo se usan todos los datos originales de entrenamiento del VAE y todos los datos generados para entrenar el modelo. Con este modelo se pretende verificar definitivamente si los datos sintéticos permiten entrenar mejor a los modelos.

De aquí se extraerán los resultados para realizar el posterior análisis.

4.Análisis de resultados

En este apartado se expondrá y analizará los resultados obtenidos siguiendo el procedimiento que se ha descrito en el apartado anterior.

4.1 Análisis de resultados de los modelos generativos

Lo primero que se hace es el entrenamiento de todas las combinaciones entre modelos generativos y grupos de datos. Los parámetros de entrenamiento se han mantenido constantes en todos los modelos y son los que se muestran en la Tabla 6.

Parámetro	Valor
epochs	20
batch_size	128
validation_split	0.2
shuffle	True
callbacks	EarlyStopping
es_patience	5

Tabla 6 Parámetros de entrenamiento VAE

Se han escogido veinte épocas ya que se ha visto que son épocas suficientes para que los entrenamientos de los modelos converjan. A pesar de ello, se ha decidido utilizar la técnica del *early stopping* para evitar el *overfitting*.

Se decide hacer una separación de los datos de entrenamiento para la validación del veinte por ciento y se decide usar un *batch size* de 128 muestras. Finalmente se decide poner el *shuffle* a True porque gracias a esto se consigue mejorar el entrenamiento.

Se recuerda que la notación de los modelos es las siguiente:

- Modelo 1: Modelo con una capa de LSTM de 64 unidades.
- Modelo 2: Modelo con una capa de LSTM de 80 unidades.
- Modelo 3: Modelo con dos capas de LSTM de 64 y 32 unidades.

Y la notación de los grupos de datos es la siguiente:

- T: Grupo de datos que solo está formado por las variables categóricas temporales.
- T-D: Grupo de datos formado por las variables categóricas temporales y descriptivas.
- T-D-N: Grupo de datos formado por todas las variables categóricas.

En las tablas 7, 8 y 9 se muestra el resultado del entrenamiento para los distintos VAE.

Training time (min)			
Modelo \ Datos	T	T-D	T-D-N
Modelo 1	11,9	11,2	10,38
Modelo 2	22,3	15,2	16,1
Modelo 3	22	21,3	13,2

Tabla 7 Tiempo de entrenamiento VAE

Train loss			
Modelo \ Datos	T	T-D	T-D-N
Modelo 1	2,387	1,778	1,842
Modelo 2	2,183	1,765	1,765
Modelo 3	2,804	1,48	2,495

Train r2 (mean/std)			
Modelo \ Datos	T	T-D	T-D-N
Modelo 1	0,952/0,042	0,961/0,031	0,960/0,036
Modelo 2	0,957/0,041	0,961/0,030	0,961/0,038
Modelo 3	0,937/0,0719	0,968/0,027	0,945/0,045

Tabla 8 Resultados del VAE para los datos de entrenamiento

Test loss			
Modelo \ Datos	T	T-D	T-D-N
Modelo 1	2,414	1,625	2,094
Modelo 2	2,27	1,634	1,683
Modelo 3	2,554	1,818	2,451

Test r2 (mean/std)			
Modelo \ Datos	T	T-D	T-D-N
Modelo 1	0,955/0,068	0,967/0,025	0,957/0,048
Modelo 2	0,957/0,071	0,966/0,027	0,966/0,027
Modelo 3	0,948/0,066	0,963/0,041	0,950/0,045

Tabla 9 Resultados del VAE para los datos de entrenamiento

Para cada modelo se han obtenido cinco parámetros del resultado de su entrenamiento: *train loss*, *test loss*, *r^2 train*, *r^2 test* y tiempo de entrenamiento. Los valores de estos parámetros permiten comparar los modelos entre ellos.

A simple vista parece que hay dos combinaciones de modelo-grupo de datos mejores que los demás por los resultados generales que han obtenido en todos los parámetros de entrenamiento, como son el Modelo 1 con los grupos de datos T-D y T-D-N. El modelo 2 con todos los grupos de datos también ha obtenido buenos resultados en general, aunque no estén a la altura de los del modelo 1. Por el contrario, el modelo 3 parece que no termina de obtener tan buenos resultados como los demás.

Estos resultados parecen contradictorios al que diría la lógica pues el modelo 3 es el que tiene mayor capacidad, pero si se mira el proceso de entrenamiento, nos damos cuenta que esta gran capacidad es la que le está dificultando el proceso de entrenamiento. En la ilustración 18 se puede ver la evolución de la función de pérdida a lo largo de las épocas para el modelo 3 y se puede ver que parece que el modelo es capaz de continuar aprendiendo, pero este proceso es muy lento. Por lo tanto, se puede concluir que el modelo 3 esta sobre capacitado.

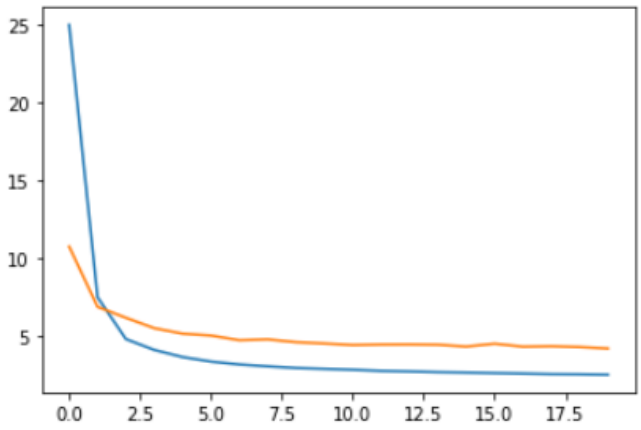

Ilustración 18 Evolución a lo largo de las épocas de la función de perdida para el modelo 3

Después de haber analizado los resultados visualmente se decide utilizar un criterio subjetivo para poder escoger las mejores combinaciones de modelo-grupo de datos. Se decide optar por la matriz de decisiones como criterio subjetivo, ya que dicha matriz permite escoger de manera sistemática y ordenada entre varias opciones en función de unos factores a los que se les aplica unos pesos.

En nuestro caso, las opciones de la matriz de decisiones son las distintas combinaciones entre modelo-grupo de datos, mientras que los factores son los parámetros que se han usado para la evaluación de los modelos. Para empezar a realizar la matriz de decisiones se debe concretar tres elementos para cada factor.

- El primer elemento que se debe tener en cuenta es el peso que se le asigna a cada factor, que va a depender de la importancia que se le dé al factor. En nuestro caso se pueden ver los pesos que se han fijado para cada factor en la fila de Peso de la matriz de decisiones que se muestra

en la tabla 10. Se ha decidido dar una importancia alta a las funciones de pérdida y al tiempo de entrenamiento, una importancia media a las medias del r^2 y una importancia baja a las desviaciones estándar del r^2.

- El segundo elemento que se debe tener en cuenta es que todos los factores menos la media del r^2 se pretenden minimizar. Para poder trasladar esta característica de los factores a la matriz de decisiones en vez de usar directamente el valor que se ha obtenido, se usa la inversa del valor.
- El tercer elemento que se debe tener en cuenta es el escalado de los factores, ya que no todos los factores son del mismo orden. Para solucionar este problema lo que se hace es escalar todos los factores a diez. Este escalado se realiza creando un valor de escala que se obtiene de dividir 10 entre el valor máximo de este factor.

Finalmente se obtiene la Puntuación total de cada modelo con la ecuación 3.

$$Puntución\ total = \sum_{i=1}^{No\ factores} (Peso\ factor * valor\ de\ escala\ factor * valor\ factor)$$

Ecuación 3 Puntuación de cada modelo en la matriz de decisiones

En la tabla 10 se muestra la matriz de decisiones. Con la que se puede sacar conclusiones más cuantificables.

	Train loss	Test loss	Train r2 mean	Train r2 std	Test r2 mean	Test r2 std	Training time	Total
Peso	9,00	10,00	7,00	5,00	7,00	5,00	9,00	
Escala (10)	14,80	16,34	10,33	0,27	10,34	0,25	103,80	
Modelo 1 (T)	0,41	0,42	0,95	23,81	0,96	14,71	0,08	390,64
Modelo 1 (T-D)	0,62	0,56	0,96	32,26	0,97	40,00	0,09	478,92
Modelo 1 (T-D-N)	0,48	0,54	0,96	27,78	0,96	20,83	0,10	439,20
Modelo 2 (T)	0,46	0,44	0,96	24,39	0,96	14,08	0,04	363,12
Modelo 2 (T-D)	0,57	0,61	0,96	33,33	0,97	37,04	0,07	454,79
Modelo 2 (T-D-N)	0,57	0,59	0,96	26,32	0,97	37,04	0,06	448,44
Modelo 3 (T)	0,36	0,39	0,94	13,91	0,95	15,15	0,05	341,41
Modelo 3 (T-D)	0,68	0,55	0,97	37,04	0,96	24,39	0,05	426,08
Modelo 3 (T-D-N)	0,40	0,41	0,95	22,22	0,95	22,22	0,08	387,85

Tabla 10 Matriz de decisiones

Se ha podido observar que la mejor combinación entre modelo y grupo de datos es el “Modelo 1 (T-D)” que ha obtenido una puntuación de 478,92, seguido de cerca por el “modelo 2 (T-D)” con una puntuación de 454,79 y el “modelo 2 (T-D-N)” con una puntuación de 448,44.

Si se fija solo en los datos se puede ver como los que obtienen mejor resultado son los (T-D), muy seguidos de cerca por los (T-D-N). Es curioso ver este comportamiento ya que en el segundo caso se dispone de más información y por lo tanto los modelos deberían obtener mejores resultados.

Por otra parte, cuando los modelos solo han usado los datos (T) no han obtenido muy buenos resultados comparados con los demás grupos de datos, pero hay que tener en cuenta que solo hay tres variables temporales y cuatro variables descriptivas, y estas últimas solo aportan una mejora de unos 100 puntos de media. Quizás sería interesante realizar un estudio solo con las variables descriptivas y sin las variables numéricas para ver qué puntuación obtienen, aunque se supone que sería mucho más baja que con solo las variables temporales ya que en los primeros apartados se ha visto que estas segundas están mucho más correlacionadas con las variables objetivo que las primeras.

Si ahora se pasa a ver los resultados de los modelos se puede observar cómo el mejor modelo es el modelo 1 seguido muy de cerca por el modelo 2; por el contrario, el modelo 3 no obtiene muy buenos resultados. Estos resultados son los que se ha comentado antes de realizar la matriz de decisiones, y que se ha concluido que este comportamiento viene dado por un sobredimensionamiento del modelo 3 y en menor medida del modelo 2.

Visto esto se ha decidido entrenar un modelo de prueba con 32 unidades en las capas ocultas de LSTM y los resultados para este modelo han estado muy por debajo de los tres modelos que se ha presentado en este trabajo. Visto esto, se decide continuar con los modelos obtenidos ya que todo indica que el modelo 1 es el que puede obtener mejores resultados de todas las combinaciones posibles.

Para el siguiente paso del procedimiento del trabajo, la generación de datos se decide continuar con tres combinaciones de modelos con grupos de datos:

1. Modelo 1 (T-D)
2. Modelo 2 (T-D-N)
3. Modelo 1 (T)

Se ha escogido la primera combinación porque es el que ha obtenido mejor puntuación en la matriz de decisiones. No se ha seleccionado la combinación que ha obtenido los segundos mejores resultados en la matriz de decisiones por su gran similitud con la que ha obtenido la mejor puntuación, se prefiere diversificar.

La combinación entre modelo y grupo de datos que sí se ha seleccionado es el tercer modelo con más puntuación en la matriz de decisiones (Modelo 2 (T-D-N)). Este segundo modelo ya presenta más

diferencias respecto al primer modelo seleccionado y por lo tanto facilitará su comparación en los apartados posteriores del trabajo.

Finalmente, se decide escoger como tercera combinación entre modelo y grupo de datos el Modelo 1 (T). Se ha optado por su elección por el hecho de que es interesante estudiar un modelo que solo use las variables categóricas temporales, ya que, como se ha comentado previamente, todas las series temporales disponen de estas variables categóricas. Esta combinación puede servir como referencia a otros conjuntos de datos que solo tengan variables categóricas temporales.

En las Ilustraciones 19 y 20 se muestra gráficamente la reconstrucción para analizar, respectivamente, los datos de entrenamiento vs los datos originales y los de test respecto los datos originales. Estas gráficas sobre las reconstrucciones se han obtenido del Modelo 1 (T-D), es decir, el que ha obtenido mejores resultados.

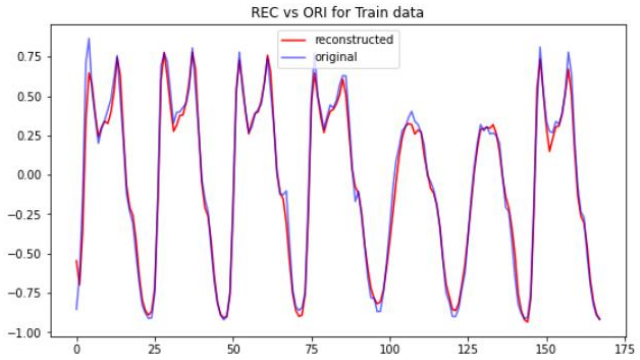

Ilustración 19 Reconstrucción de los datos de entrenamiento

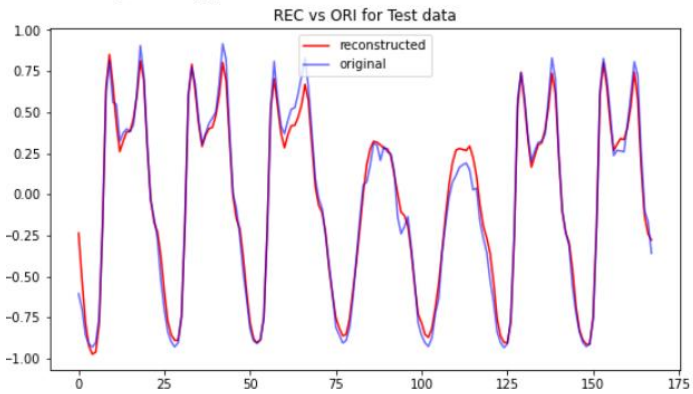

Ilustración 20 Reconstrucción de los datos de test

Gracias a estos gráficos se ha podido comprobar gráficamente que la reconstrucción también es correcta.

4.2 Generación de nuevos datos

Para realizar la reconstrucción de los datos lo primero que se hace es estudiar el espacio latente, cuyo resultado se ha avanzado en el apartado *3.1 Modelos generativos* (ver Ilustraciones 18 y 19). Se ha visto que, después del entrenamiento, el espacio latente sigue una distribución gaussiana uniforme, que es lo que cabe esperar y que permite la generación de nuevos datos.

Antes de generar los nuevos datos se debe comprobar si el espacio latente tiene alguna tendencia en función de alguna variable categórica. Esta tendencia no se debería esperar ya que el espacio latente debe ser aleatorio. Para comprobarlo se ha realizado un análisis visual, para cada variable, el cual se muestra en la Ilustración 21.

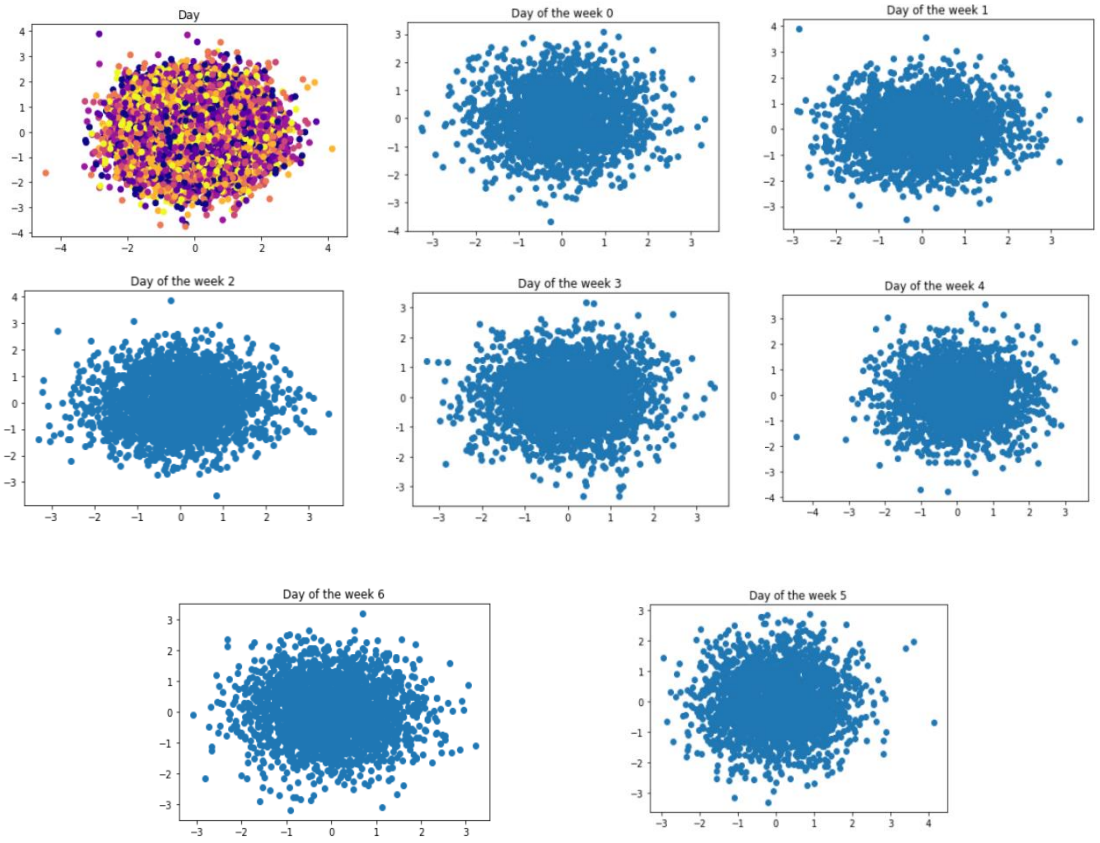

Ilustración 21Distribución del espacio latente para la variable categórica Day

En este conjunto de imágenes solo se muestra para la variable categórica *Dia*, que es una de las que tenía mas influencia en la variable objetivo, y se puede comprobar como la distribución es bastante uniforme en todos los valores sin que siga ninguna dispersión especial ni ninguna tendencia.

Entonces los modelos generativos estan bien entrenados para realizar la reconstrucción de los datos y para ordenar el espacio latente, por lo tanto se puede generar datos a partir de ellos esperando obtener un buen resultado.

En este punto se realiza la reconstrucción de los datos y se comparan estos gráficamente con los datos generados para ver si visualmente se puede empezar a ver una similitud entre ellos. En la Ilustración 22 se muestra dos secuencias aleatorias en que se compara los datos generados y los originales, recordemos que los dos han tenido como entradas las mismas variables categóricas.

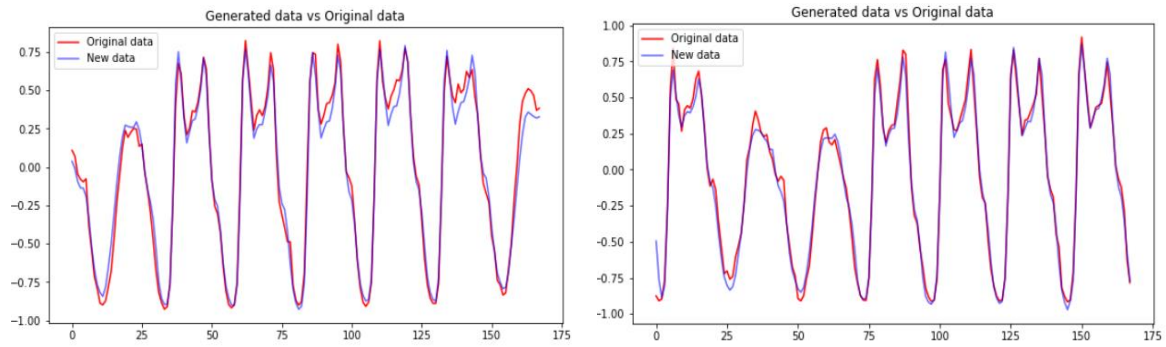

Ilustración 22 Datos generados vs datos reales

Se observa que las tendencias son muy parecidas, pero vemos comportamientos puntuales un poco distintos entre los dos tipos de datos. Este hecho es lo que se esperaba, ya que estamos reproduciendo unos datos con las mismas propiedades, pero no se pretende calcarlos. Por lo tanto, la generación de datos parece bastante bien hecha y no se encuentra indicios de lo contrario.

Es importante comprobar la unión entre dos muestras para ver que los datos generados estén haciendo bien estas uniones ya que puede ser que no mantengan la cohesión entre muestras. Para comprobar este suceso se concatena dos muestras seguidas, como se muestra en la Ilustración 23, y también se grafían dos muestras consecutivas de los datos generados, como se muestra en la Ilustración 24.

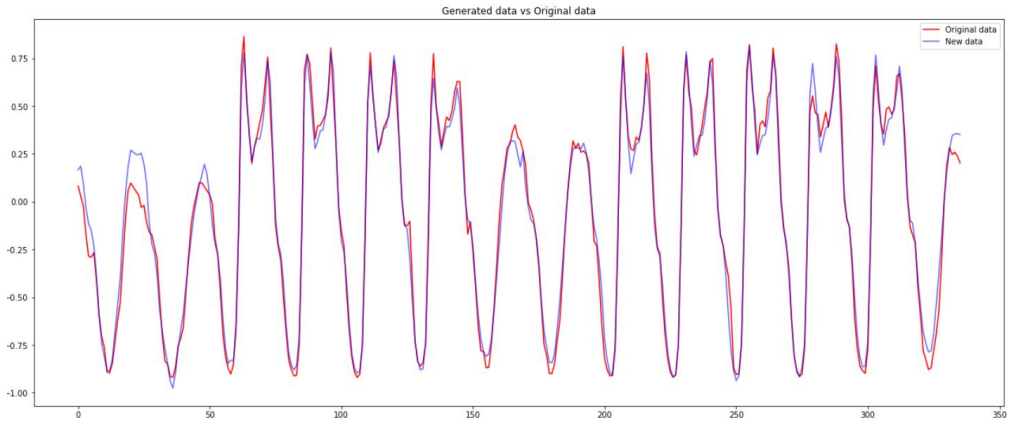

Ilustración 23 Unión entre dos muestras de datos generados y reales

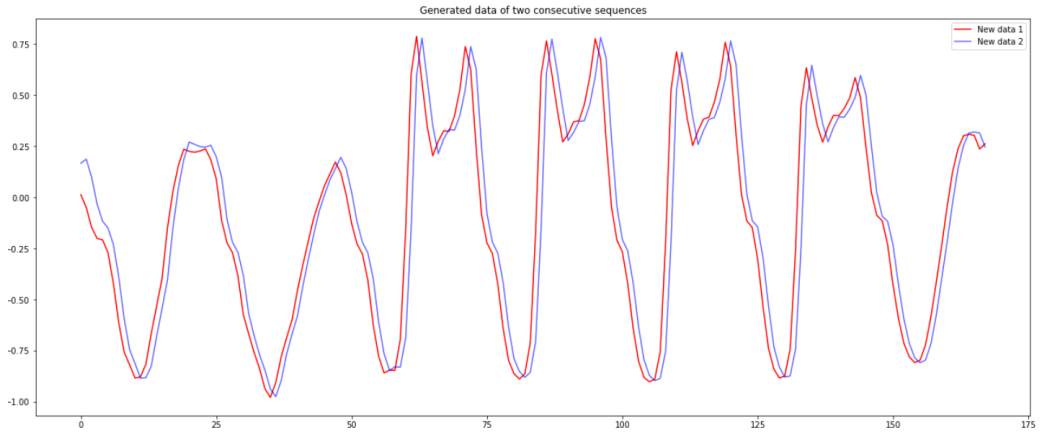

Ilustración 24 Dos secuencias consecutivas de datos generados

En la primera imagen, podemos observar como la unión entre dos muestras no presenta ningún problema significativo a simple vista y con la segunda imagen se comprueba que la serie temporal la reproduce idénticamente. Por lo tanto, se puede concluir que a simple vista parece que los nuevos datos se han generado bien y se puede pensar que se pueden utilizar para la técnica de la *data augmentation*.

4.3 Resultados de generación de datos

Una vez se ha visto que los datos sintéticos que se ha generado parecen adecuados para el *data augmentation*, se pasa a hacer la comprobación práctica de esto. Esta comprobación se hace mediante modelo de *Autoencoders* y como se ha descrito en el apartado *3.3 Comprobación de la calidad de los resultados.*

Se recuerda que este estudio se pretende hacer para tres combinaciones de modelos y grupos de datos. La primera combinación era el Modelo 1 (T-D). Los resultados del entrenamiento de los Autoencoders obtenidos por los datos sintéticos generados por esta combinación entre modelo y grupo de datos se encuentran en la tabla 11. En la tabla 13 se encuentra los resultados para los datos generados por el Modelo 2 (T-D-N). Finalmente, los resultados obtenidos por los datos sintéticos generados por el Modelo 1 (T) se encuentran en la tabla 15.

Los valores que se ha monitorizado para el estudio del entrenamiento de cada autoencoder son los siguientes:

- Loss (Entrenamiento): Función de perdida (MSE) para los datos de entrenamiento del modelo generativo.
- Loss (Test): Función de perdida (MSE) para los datos de test del modelo generativo.
- Loss (Generados): Función de perdida (MSE) para los datos generados.

- Tiempo de entrenamiento (s): tiempo que ha tardado el Autoencoder en completar el entrenamiento, en segundos.

En las tablas que se muestran a continuación sobre los resultados del entrenamiento de los Autoencoders se ha marcado en verde las casillas con los datos que se ha usado como entrenamiento. Esto se hace como recordatorio.

El primer modelo que se analiza es el Modelo 1 (T-D), Tabla11. El autoencoder A1 que solo ha entrenado con los datos de entrenamiento del modelo generativo ha obtenido los peores resultados ya que están un orden por debajo de los demás. Se observa como en los siguientes Autoencoders, que empiezan a entrenar con datos generados, la función de pérdida de todos los datos va disminuyendo.

Modelo 1 (T-D)				
Entrenamiento	Loss (Entrenamiento VAE)	Loss (Test VAE)	Loss (Generados)	Tiempo de entrenamiento (s)
AE1	8,40E-04	7,76E-04	8,81E-04	20
AE2	4,39E-05	1,62E-05	7,91E-05	18
AE3	1,46E-04	2,57E-04	5,34E-05	22
AE4	1,37E-05	1,85E-06	3,04E-05	28

Tabla 11 Resultado de las reconstrucciones de los Autoencoders para el modelo generativo 1 (T-D)

Al hacerse complicado estudiar los resultados con los valores obtenidos, se transforman dichos valores a incrementos respecto al AE1. Se decide usar incrementos logarítmicos, ecuación 4, pues como se ha visto los órdenes entre valores son distintos. Estos incrementos están hechos respecto el AE1 y de tal forma que un aumento porcentual significa una disminución en valor, que comporta una mejora de resultado. En la tabla 12 se muestra los resultados de esta transformación.

$$\Delta\% = \frac{\log(x) - \log(x_0)}{\log(x_0)}$$

Ecuación 4 Incremento logarítmico porcentual

Modelo 1 (T-D)				
Entrenamiento	Loss (Entrenamiento VAE)	Loss (Test VAE)	Loss (Generados)	Tiempo de entrenamiento (s)
AE1	0,00%	0,00%	0,00%	0,00%
AE2	41,68%	54,03%	34,26%	-3,52%
AE3	24,71%	15,45%	39,85%	3,18%
AE4	58,12%	84,33%	47,86%	11,23%

Tabla 12 Incrementos porcentuales del resultado de las reconstrucciones de los Autoencoders para el modelo generativo 1 (T-D)

Ahora sí que se puede observar mejor el comportamiento que se ha comentado anteriormente. Se puede ver como en todos los casos existe una mejora respecto al entrenamiento del AE1. El valor que nos proporciona una mejor conclusión es la Loss (Test VAE) ya que estos son datos originales que en ninguno de los casos el modelo los ve durante el entrenamiento. El valor de Loss (Test VAE) sube en todos los casos.

Se puede observar cómo entrenando el modelo solo con datos generados se obtiene una mejora muy significativa en los datos reales (entrenamiento y test), es decir entrenando con solo con los datos sintéticos el Autoencoder es capaz de reconstruir mejor los datos de test que solo entrenando con los datos de entrenamiento. Gracias a esto se puede ver las buenas propiedades que tienen los datos generados.

Los resultados son más prometedores si nos fijamos en el AE4 con el que se realiza lo que se debería hacer si se hubiera usado la técnica de la data augmentation para el conjunto de datos. Los valores resultados de este Autoencoder suponen una mejora del 84% en los datos de Test.

Finalmente se puede observar que con los datos generados el tiempo disminuye, esto se puede deber a que los datos generados no tienen valores anómalos, son muy constantes. Si que en el AE4 aumenta el tiempo de entrenamiento, pero esto es debido a que tiene que entrenar con el doble de datos, el incremento es despreciable si se tiene en cuenta este factor.

Por lo tanto, con este modelo se ha visto que los datos generados refuerzan el entrenamiento de los modelos neuronales. Veamos si este comportamiento se mantiene con las otras combinaciones de modelos y datos.

Modelo 1 (T)				
Entrenamiento	Loss (Entrenamiento VAE)	Loss (Test VAE)	Loss (Generados)	Tiempo de entrenamiento (s)
AE1	1,16E-03	1,65E-03	5,18E-04	18
AE2	2,17E-04	3,65E-04	4,26E-05	16
AE3	2,18E-04	3,66E-04	4,32E-05	19
AE4	2,30E-05	5,38E-05	1,26E-06	21

Tabla 13 Resultado de las reconstrucciones de los Autoencoders para el modelo generativo 1 (T)

Modelo 1 (T)				
Entrenamiento	Loss (Entrenamiento VAE)	Loss (Test VAE)	Loss (Generados)	Tiempo de entrenamiento (s)
AE1	0,00%	0,00%	0,00%	0,00%
AE2	24,80%	23,55%	33,03%	-4,08%
AE3	24,73%	23,50%	32,83%	1,87%
AE4	58,00%	53,42%	79,56%	5,33%

Tabla 14 Incrementos porcentuales del resultado de las reconstrucciones de los Autoencoders para el modelo generativo 1 (T)

Modelo 2 (T-D-N)				
Entrenamiento	Loss (Entrenamiento VAE)	Loss (Test VAE)	Loss (Generados)	Tiempo de entrenamiento (s)
AE1	1,91E-03	2,60E-03	1,60E-03	22
AE2	2,65E-04	4,36E-04	1,62E-04	20
AE3	6,96E-05	3,49E-05	1,10E-04	19
AE4	3,96E-06	2,90E-07	2,56E-05	25

Tabla 15 Resultado de las reconstrucciones de los Autoencoders para el modelo generativo 2 (T-D-N)

Modelo 2 (T-D-N)				
Entrenamiento	Loss (Entrenamiento VAE)	Loss (Test VAE)	Loss (Generados)	Tiempo de entrenamiento (s)
AE1	0,00%	0,00%	0,00%	0,00%
AE2	31,55%	30,00%	35,51%	-3,08%
AE3	52,90%	72,42%	41,52%	-4,74%
AE4	98,69%	152,90%	64,15%	4,14%

Tabla 16 Incrementos porcentuales del resultado de las reconstrucciones de los Autoencoders para el modelo generativo 2 (T-D-N)

Se ha podido observar como el comportamiento que se ha comentado con el primer modelo generativo se mantiene con los otros dos. No se puede observar ninguna diferencia notable en cuanto a comportamiento individual. Por lo tanto, se llega a las mismas conclusiones que en el primer caso: Los datos sintéticos que se ha generado refuerzan el entrenamiento de los modelos neuronales.

Si se pasa a comparar los tres modelos entre ellos se decide fijarse únicamente en el AE4 que es el que nos proporciona la información de si se ha realizado bien la data aumentación. En la tabla 17 se muestran los valores.

Modelo generativo	Entrenamiento	Loss (Entrenamiento VAE)	Loss (Test VAE)	Loss (Generados)	Tiempo de entrenamiento (s)
Modelo 1 (T-D)	AE4	58,12%	84,33%	47,86%	11,23%
Modelo 1 (T)	AE4	58,00%	53,42%	79,56%	5,33%
Modelo 2 (T-D-N)	AE4	98,69%	152,90%	64,15%	4,14%

Tabla 17 Comparación de los modelos generativos para el data aumentación

Se puede ver que el Modelo 2 (T-D-N) que como modelo generativo hacía peor la reconstrucción que el Modelo 1 (T-D) y ahora es el que hace mejor la data augmentation superando en todos los valores de loss a los otros modelos generativos. Este modelo ha llegado a obtener una mejora del 152,9 %.

El modelo 1 (T-D) es el que obtiene peores resultados en la reconstrucción como modelo generativo y parece que esta tendencia continua con los datos que ha generado, ya que han sido los que han obtenido peor resultado. Aun así, ha obtenido una mejora constante y bastante significativa respecto los valores reales.

Parece que la tendencia para los datos generados es la que se esperaba en los modelos generativos, cuantas más variables categóricas tenga de entrada el modelo mejor resultado obtiene.

En cuanto al objetivo global, generar nuevos datos que permitan entrenar modelos de redes neuronales con mayor eficiencia, se puede decir sin ninguna dudad de que se ha alcanzado, pues todos los

modelos presentan una mejora significativa en sus entrenamientos si se usa los datos generados junto con los datos de entrenamiento.

Cabe decir que los modelos de redes neuronales con los que se ha probado son muy básicos, pero es de suponer que se pueda hacer la extrapolación a modelos más complejos con porcentajes de mejora parecidos a los que se ha obtenido.

Análisis visual de la reconstrucción

Visto que los datos sintéticos generados son capaces de mejorar el resultado, a continuación, se muestra esta mejora gráficamente.

Se usa el Modelo 2 (T-D-N) que ha obtenido los mejores resultados. En la Ilustración 25 se muestra la evolución de la función de pérdida en función de las épocas, a la izquierda el AE1 y a la derecha el AE2. Se observa como en los dos casos el resultado ha convergido, por lo tanto, no son susceptible de mejora.

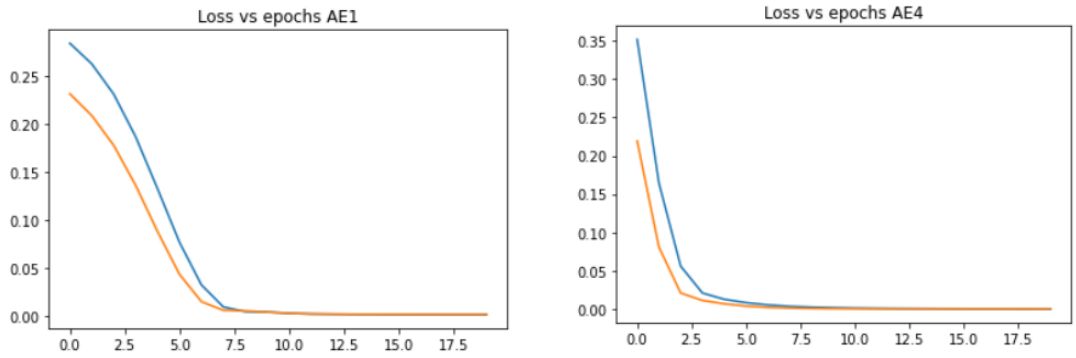

Ilustración 25 Evolución de la función de perdida para los modelos AE1 y AE2 del modelo generativo 2 (T-D-N)

Una vez se ha visto que no tiene margen de mejora se pasa a ver la reconstrucción que hace cada uno de los modelos para cada tipo de dato, mostrándose en la Ilustración 26 estas reconstrucciones, a la izquierda la que hace el AE1 y a la derecha la que hace el AE4.

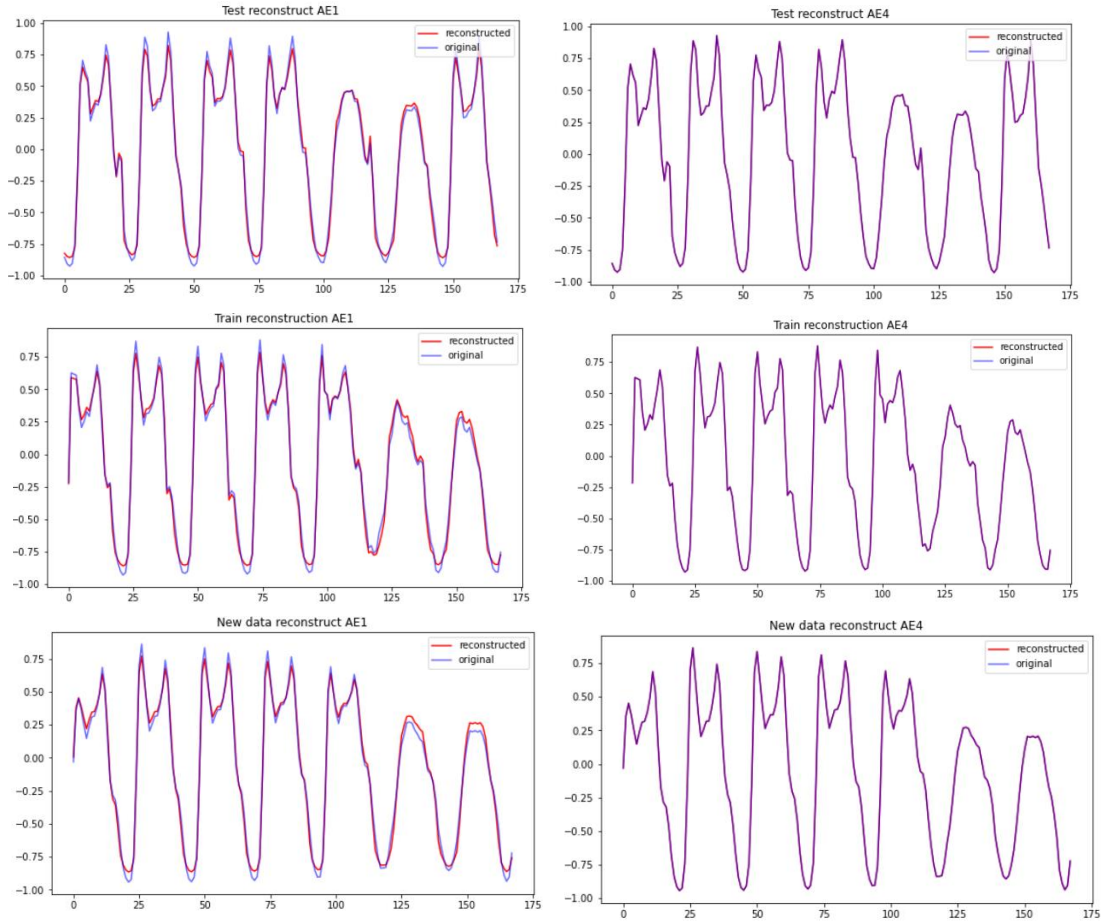

Ilustración 26 Reconstrucción de los AE1 y AE4 para el modelo generativo 2 (T-D-N)

Se ha podido observar que la reconstrucción mejora gráficamente si se compara el modelo entrenado sin usar los datos generados y entrenado usando los datos generados. Por lo tanto, se llega a la misma conclusión tanto analíticamente como gráficamente.

5. Líneas de futuro

Una vez se ha llegado en este punto en que se puede responder a los objetivos que se ha marcado este trabajo vale la pena mirar las posibilidades de continuación de este proyecto.

La primera forma para continuar con este trabajo sería el uso de otros tipos modelos generativos para la *data augmentation* de datos de serie temporal, como es el caso de los GAN. Para poder comparar los resultados obtenidos mediante los dos modelos generativos y así poder analizar cual obtiene mejores resultados.

También, vistos los buenos resultados que ha obtenido los modelos generativos que se ha usado en este trabajo, se ha pensado que éstos se podrían usar con fines distintos para los que se han usado en este trabajo: el uso del modelo generativo como modelo predictivo o el uso del modelo generativo como sustituidor de NaN. Estos dos se explican a continuación por su poca varianza respecto lo que se ha realizado en este trabajo.

Modelo predictivo

El primer fin con el que se puede reciclar el uso de los modelos generativos usados en este trabajo es el de poder hacer predicciones. Para hacerlo se debería entrenar el modelo de la misma forma que se ha hecho. Sería recomendable entrenarlo con el máximo número de variables categóricas, pero estas variables se deben limitar a los valores que se pueda obtener con facilidad.

Si se vuelven a observar los datos que se ha generado anteriormente y se escalan para que obtengan la dimensión original se puede observar que hay muestras negativas, Ilustración 27. Este problema de reconstrucción se debe corregir ya que no tiene sentido un volumen de tráfico negativo.

	holiday	weather_main	weather_description	traffic_volume
2016-05-01 00:00:00	None	Rain	light rain	2197
2016-05-01 01:00:00	None	Clouds	overcast clouds	1177
2016-05-01 02:00:00	None	Clouds	overcast clouds	425
2016-05-01 03:00:00	None	Clouds	overcast clouds	-88
2016-05-01 04:00:00	None	Drizzle	drizzle	-221

Ilustración 27 Datos generados escalados

Posteriormente, una vez se haya corregido esto, para predecir valores de días futuros, primero se deben obtener las variables categóricas que se predigan para esos días (las temporales son sencillas

de obtener y las demás estarían al alcance también si la fecha que se desee predecir esté cerca del día presente) y usar el modelo generativo para predecir los valores de volumen de tráfico.

Sustitución de valores NaN

La segunda forma de reciclar el modelo generativo que utiliza este trabajo consiste en utilizarlo como método de sustitución de los valores NaN. Este método pretende sustituir los valores NaN por valores predichos. Dentro de este grupo se debe diferenciar entre dos casos: agrupación muchos valores NaN consecutivos, valores NaN aislados en el conjunto de datos. La única diferencia entre los dos casos es las variables categóricas de entrada que se le debe proporcionar al modelo:

- Si se está en el caso de que existe un valor NaN aislado lo que se puede hacer es propagar las variables categóricas del periodo anterior o posterior y predecir el valor mediante el modelo generativo para la observación actual.
- Si por el contrario se está en el caso de que se encuentra muchos valores NaN consecutivos lo más recomendable seria usar solo las variables categóricas temporales como entradas del modelo.

6. Plan temporal

El proyecto se ha realizado desde marzo de 2021 hasta junio de 2021. Los distintos apartados realizados durante este trabajo se muestran en el diagrama de Gantt de la Ilustración 28.

El código de colores es el siguiente:

- Azul: Búsqueda de información
- Marrón: Trabajo práctico
- Verde: Redacción

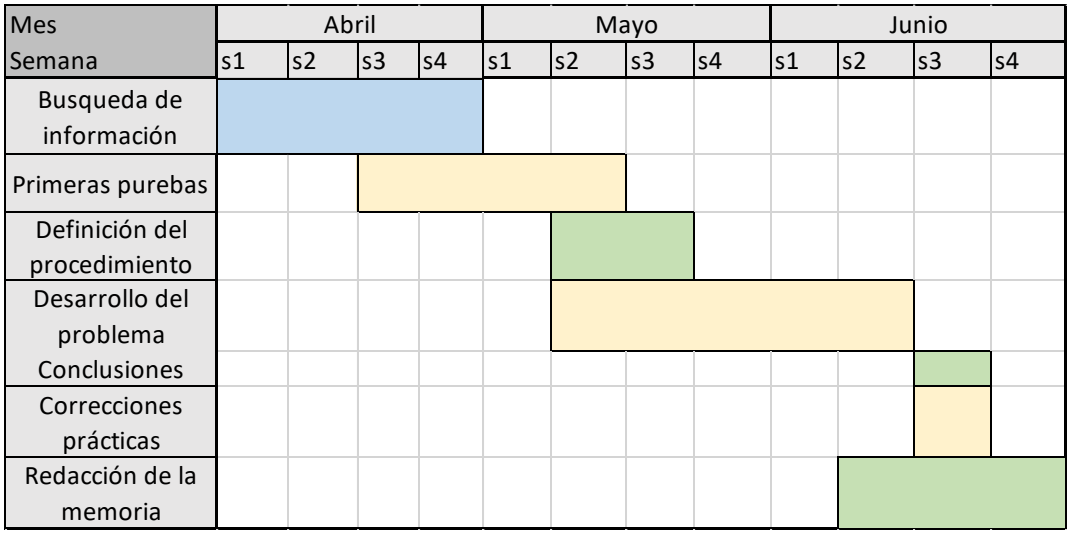

Ilustración 28 Diagrama de Gantt

7. Estudio ambiental

El impacto ambiental de este trabajo es el que haya podido generar el uso del ordenador portátil, único hardware que se ha utilizado en este trabajo. El uso del ordenador puede repercutir ambientalmente en dos aspectos: el impacto ambiental que genere la energía utilizada por el ordenador a lo largo de este trabajo y el impacto ambiental que genere el ordenador como residuo.

Se considera un proyecto de impacto nulo, suponiéndose que la electricidad utilizada se ha generado lo más limpiamente posible y sabiendo que se puede reciclar un 93%, teniendo en cuenta que no se ha tenido que imprimir ningún material ni se ha tenido que realizar ningún desplazamiento con el fin de realizar este proyecto.

8. Estudio económico

En el presente apartado se ha tratado de realizar un cálculo aproximado del presupuesto del proyecto. Para realizarlo se ha tenido en cuenta cuatro factores: las horas dedicadas por el autor, las licencias de los programas utilizados, el coste de los equipos utilizados y el coste de la energía consumida por los equipos.

Las horas que ha dedicado el autor en este trabajo son las que se muestran en la tabla 18. Para el precio unitario de las horas de trabajo se usa el de la media española que se encuentra actualmente en 14 €/hora.

Concepto	Horas dedicadas [h]
Búsqueda de información	20
Experimentación preliminar	75
Diseño de la estructura del trabajo	5
Realización de la parte práctica del trabajo	90
Análisis de resultados y conclusiones	50
Redacción de la memória	80
Total	**320**

Tabla 18 Horas dedicadas al trabajo

Para la realización de este trabajo se ha usado dos programas: Microsoft Excel y Microsoft Word. Los dos programas están incluidos en el paquete básico de Microsoft que tiene un precio de 69 €/año. Se debe tener en cuenta que el trabajo ha durado 0,25 años.

Para la realización del programa se ha utilizado *colab*, un servicio en la nube basado en las Notebooks de *Jupyter*. Esta herramienta es gratuita.

Partimos de la vida útil media de cinco años que tienen los ordenadores, y teniendo en cuenta que se ha hecho uso del mismo tres meses equivalentes a 0,25 años, siendo su coste de adquisición el de 650 €, comporta la amortización parcial de dicha herramienta de trabajo. Se supone que no se hace ninguna reparación al ordenador en toda su vida útil.

Respecto a la energía consumida por el ordenador ascendente a 180 W, aplicando a las horas que se ha usado el ordenador y que se ha estado consumiendo electricidad son las mismas que las horas que se le ha dedicado al trabajo, por lo que se tiene en cuenta un coste medio de la electricidad de 0,14 €/kWh.

En la tabla 19 se resume todo lo anteriormente descrito.

Concepto	***Unidades***	***Precio unitario [€/unidad]***	***Precio total [€]***
Horas de trabajo	320	14	4480
Licencias de programas	0,25	69	17,25
Coste de equipos	0,25	130	32,5
Coste de electricidad	320	0,0252	8,06
Total			**4537,81**

Tabla 19 Presupuesto del trabajo

Añadiendo el impuesto correspondiente (IVA) al tipo legal actualmente vigente del 21% el presupuesto total del trabajo es de 5.490,75 € (IVA incluido).

9. Conclusiones

Este trabajo ha planteado tres objetivos:

1. Estudio de la influencia de las variables de los datos temporales:

Para llegar a analizar este objetivo, primero se ha visto que existen tres grupos diferentes de variables en los datos de serie temporal: variables temporales, variables categóricas y variables numéricas. Se ha visto que normalmente el objeto de estudio son las variables numéricas y que las que no sean objetivo de estudio es mejor factorizarlas para que sean utilizadas como entradas para los modelos generativos.

Posteriormente, para ver cómo afectaba cada una de estas variables al resultado del modelo, se ha utilizado el mismo modelo pero con distintas entradas. De los resultados, se ha podido ver que las variables temporales son las que aportan más información y las que deberían ser las últimas en sacarse del entrenamiento de modelos de redes neuronales, seguidas por las variables categóricas y finalmente por las numéricas.

Por lo tanto, se ha conseguido alcanzar este objetivo, ya que se ha podido ver cómo influyen las variables en el resultado de los entrenamientos de las redes neuronales, además de proponerse un método para medir esta influencia de forma cuantitativa (la matriz de decisiones).

2. Estudio e implementación de modelos generativos:

Para este segundo objetivo se ha implementado distintos modelos generativos tipo VAE, obteniendo buenos resultados en todos ellos. Antes de realizar la implementación práctica de los modelos se han estudiado teóricamente.

Se ha creados tres modelos generativos con distintas estructuras internas para ver cómo afecta ésta al comportamiento del resultado. Con los resultados obtenidos se ha visto que para este problema en concreto solo hacía falta una *layer*. Todo y esto también se ha implementado modelos generativos tipo VAE con más de una capa oculta. La generación de los datos ha sido satisfactoria, obteniendo datos sintéticos totalmente nuevos que se parecían en apariencia a los datos originales.

Visto esto se puede decir que se ha alcanzado también este objetivo planteado al inicio del trabajo.

3. *Data augmentation* para datos de serie temporal:

Finalmente, en torno a este último objetivo ha girado todo el proyecto, ya que los dos primeros iban enfocados a éste. Por ello, después de haber generado los datos con distintos modelos y grupos de

datos se ha analizado si estos datos permiten entrenar mejor los modelos de autoaprendizaje. Para comprobarlo se ha usado modelos tipo Autoencoder.

Los resultados que se han obtenido han sido muy satisfactorios. Al hacer el entrenamiento de los modelos tipo autoencoder con la técnica de la *data augmentation* han obtenido mucho mejor resultado que con los datos originales. Por lo tanto, se ha concluido que los modelos generativos se pueden usar para la técnica de *data augmentation* para datos de serie temporal.

Es cierto que los Autoencoders no son modelos muy complejos, pero debido a sus características y a la manera en que interpretan los datos se cree que estos resultados son extrapolables a cualquier otro tipo de modelo neuronal.

Por esto se puede concluir que en este trabajo se ha logrado generar datos de serie temporal sintéticos a partir de un conjunto de datos para que modelos de red neuronal puedan entrenar de forma más eficiente con ellos.

Agradecimientos

En primera instancia, quiero dar mis mayores agradecimientos al tutor que lo ha dirigido, el profesor Samir Kanaan Izquierdo, por haberme ayudado desde el inicio a llevar a cabo el proyecto, agradeciéndole la propuesta de tema y la confianza que depositó en mí.

Antes de empezar con el proyecto mis conocimientos eran nulos sobre el tema del que versa, por lo tanto, agradecerle también a él y al proyecto el interés que han despertado en mi sobre la inteligencia artificial y especialmente sobre las redes neuronales.

Finalmente, agradecer a mi familia y amigos por su soporte moral e incondicional a lo largo de este trabajo.

Bibliografía

Dataset.

[https://archive.ics.uci.edu/ml/datasets/Metro+Interstate+Traffic+Volume]

Keras.

[https://www.tensorflow.org/guide/keras?hl=es]

Numpy.

[https://numpy.org/]

Pandas.

[https://pandas.pydata.org/]

Tutoriales de ML.

[https://www.kaggle.com/]

Información sobre los VAE:

- Rocca, Joseph. Understanding Variational Autoencoders (VAEs). 2017.

[https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73]

- Cerliani, Marc. Time series generation with VAE LSTM. 2020.

[https://towardsdatascience.com/time-series-generation-with-vae-lstm-5a6426365a1c]