



EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA



TECNOLÓGICO
NACIONAL DE MÉXICO



TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE TOLUCA

DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

**DESARROLLO DE UN SISTEMA INTELIGENTE PARA LA
PLANIFICACIÓN DE PRODUCCIÓN DE SENSORES DE
PRESIÓN AUTOMOTRICES, MEDIANTE TÉCNICAS DE
APRENDIZAJE AUTOMÁTICO.**

TESIS QUE PRESENTA

JOSÉ MANUEL FLORES FLORES

NO. DE CONTROL: M23281634

COMO REQUISITO PARA OBTENER EL GRADO DE

MAESTRÍA EN CIENCIAS DE LA INGENIERÍA

DIRECTOR(A): **DRA. ERÉNDIRA RENDÓN LARA**

Hilda Moreno Saavedra
12/ agosto / 2025

METEPEC, ESTADO DE MÉXICO, MÉXICO
SEPTIEMBRE 2025



Educación
Secretaría de Educación Pública



TECNOLÓGICO
NACIONAL DE MÉXICO



Instituto Tecnológico de Toluca
División de Estudios de Posgrado e Investigación

Metepec, Edo. de México, 13/agosto/2025

DEPI-3200-442/2025

C. JOSÉ MANUEL FLORES FLORES
CANDIDATO AL GRADO DE MAESTRO
EN CIENCIAS DE LA INGENIERÍA
PRESENTE

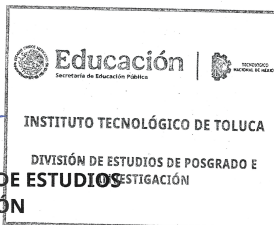
De acuerdo con los Lineamientos para la Operación de los Estudios de Posgrado en el Tecnológico Nacional de México y las disposiciones de este Instituto, habiendo cumplido con todas las indicaciones que la Comisión Revisora realizó con respecto a su trabajo de Tesis titulado **"DESARROLLO DE UN SISTEMA INTELIGENTE PARA LA PLANIFICACIÓN DE PRODUCCIÓN DE SENSORES DE PRESIÓN AUTOMOTRICES, MEDIANTE TÉCNICAS DE APRENDIZAJE AUTOMÁTICO"**, la División de Estudios de Posgrado e Investigación de este Instituto, concede la autorización para proceder a la publicación del trabajo final a través del repositorio institucional.

Sin otro particular por el momento, me despido de usted.

ATENTAMENTE

Excelencia en Educación Tecnológica®
"Educación, integridad y ciencia"

JORGE SÁNCHEZ JAIME
ENCARGADO DE LA DIVISIÓN DE ESTUDIOS
DE POSGRADO E INVESTIGACIÓN



JSJ/fvo.



2025
Año de
La Mujer
Indígena

Av. Tecnológico S/N, Col. Agrícola Bellavista, Metepec, Edo. de México,
C.P. 52149, Tels. Dirección: 7222087205, Conmut.: 7222087200
e-mail: info@toluca.tecnm.mx | tecnm.mx | toluca.tecnm.mx





Educación
Secretaría de Educación Pública



TECNOLÓGICO
NACIONAL DE MÉXICO



Instituto Tecnológico de Toluca
División de Estudios de Posgrado e Investigación

Metepec, Edo. de México, 12/agosto/2025

DEPI-3200-439/2025

JORGE SÁNCHEZ JAIME
ENCARGADO DE LA DIVISIÓN DE ESTUDIOS
DE POSGRADO E INVESTIGACIÓN
PRESENTE

Por este medio comunicamos a usted que la Comisión Revisora designada para analizar la tesis denominada **"DESARROLLO DE UN SISTEMA INTELIGENTE PARA LA PLANIFICACIÓN DE PRODUCCIÓN DE SENSORES DE PRESIÓN AUTOMOTRICES, MEDIANTE TÉCNICAS DE APRENDIZAJE AUTOMÁTICO"**, ha dictaminado que dicho trabajo reúne las características de contenido y calidad para proceder a la publicación del trabajo final a través del repositorio institucional. Lo anterior como parte de los requisitos para obtener el grado académico de Maestro en Ciencias de la Ingeniería que presenta el **C. JOSÉ MANUEL FLORES FLORES** con número de control **M23281634** para sustentar el acto de Recepción Profesional.

ATENTAMENTE

ERÉNDIRA RENDÓN LARA
DIRECTORA DE TESIS

JORGE SÁNCHEZ JAIME
REVISOR DE TESIS

FEDERICO DEL RAZO LÓPEZ
REVISOR DE TESIS

HILDA MORENO SAAVEDRA
REVISOR DE TESIS

ccp. Archivo
mcdh



2025
Año de
La Mujer
Indígena

Av. Tecnológico S/N, Col. Agrícola Bellavista, Metepec, Edo. de México,
C.P. 52149, Tels. Dirección: 7222087205, Conmut.: 7222087200
e-mail: info@toluca.tecnm.mx tecnm.mx | toluca.tecnm.mx



Dedicatoria

A mi esposa, por su amor incondicional, por acompañarme con paciencia en cada etapa de este camino y por creer en mí incluso en los momentos más difíciles, gracias por ser mi mayor motivación.

A mis padres, por enseñarme con su ejemplo el valor del esfuerzo, la disciplina y la resiliencia. Esta meta es también resultado de su guía y fortaleza.

Resumen

La presente tesis propuso el desarrollo de un sistema inteligente para la planificación de la producción de sensores de presión automotrices mediante técnicas de aprendizaje automático. Utilizó datos históricos de ventas de una empresa del sector automotriz (2018-2024), se implementó un modelo de redes neuronales recurrentes con unidades GRU (Gated Recurrent Units) para predecir la demanda de productos. La estimación de ventas es integrada con información operativa como la capacidad instalada de producción y las tasas de generación de desperdicio (scrap), con el fin de optimizar la planeación de manufactura. Los resultados demostraron que el modelo GRU ofrece una predicción robusta sin requerir la alteración de datos atípicos del periodo pandémico, siendo más eficiente al entrenarse con datos recientes (2021-2022). El sistema resultante permitió una mejor asignación de líneas de producción, minimizando desperdicios y mejorando la eficiencia operativa. Esta investigación evidenció la viabilidad del uso de inteligencia artificial en procesos industriales, aportando una solución práctica y escalable para la toma de decisiones en entornos de manufactura.

Abstract

This thesis proposed the development of an intelligent system for the production planning of automotive pressure sensors using machine learning techniques. Historical sales data from an automotive company (2018-2024) were used to implement a recurrent neural network model based on Gated Recurrent Units (GRU) for product demand forecasting. The sales estimations were integrated with operational information such as installed production capacity and scrap generation rates, aiming to optimize manufacturing planning. The results showed that the GRU model provided robust predictions without requiring modifications to the outlier data from the pandemic period, and it was more efficient when trained with recent data (2021-2022). The resulting system enabled better allocation of production lines, minimized waste, and improved operational efficiency. This research demonstrated the feasibility of applying artificial intelligence to industrial processes, offering a practical and scalable solution for decision-making in manufacturing environments.

Índice general

1. Introducción	1
1.1. Antecedentes	2
1.2. Planteamiento del Problema	4
1.2.1. Definición del problema	4
1.3. Justificación	5
1.4. Objetivos	7
1.4.1. Objetivo general	7
1.4.2. Objetivos específicos	7
1.5. Hipótesis	7
2. Fundamentos	8
2.1. Marco teórico	8
2.1.1. Rentabilidad de las empresas	8
2.1.2. Mercado vehicular y de auto partes	10
2.1.3. Proceso de cadena de valor en los sensores de presión automotrices	15
2.1.4. Descubrimiento de conocimiento en bases de datos (<i>KDD</i>)	16
2.1.5. Redes neuronales artificiales y aprendizaje profundo	26
2.2. Estado del arte	49

3. Metodología	53
3.1. Obtención de datos	53
3.2. Limpieza y normalización de datos	56
3.3. Evaluar modelo para predicción de ventas	59
3.4. Cambio de datos originales de las ventas nulas durante el covid	60
3.4.1. Código base para el tratamiento de datos	61
3.5. Selección de hiperparámetros	63
3.6. Entrenamiento del modelo por número de parte	64
3.6.1. Código base para el entrenamiento	65
3.6.2. Código para la evaluación de escenarios	70
3.7. Desarrollo de lógica que tome en cuenta la línea de producción	75
3.8. Generar un modelo que evalúe la predicción de producción	77
3.9. Generar un modelo para la predicción de producción semanal	80
4. Resultados y discusión	82
4.1. Resultados	82
4.1.1. Obtención de datos	82
4.1.2. Factibilidad de modelo GRU para la predicción de ventas	82
4.1.3. Predicción del periodo en pandemia	84
4.1.4. Predicción del periodo de pandemia con datos originales	91
4.1.5. Predicción del año 2023 con datos originales y periodo de pande- mia cambiado	93
4.1.6. Predicción de ventas para el año 2023 con un conjunto de entre- namiento menor	97
4.1.7. Análisis del cambio en la generación de <i>scrap</i> ante variaciones en la línea de producción	98

4.1.8. Generar un modelo para la predicción de producción semanal . .	100
4.2. Discusión	102
5. Conclusiones y recomendaciones	104
5.1. Conclusiones	104
5.2. Recomendaciones	105
Referencias	108

Índice de figuras

2.1. Producción vehicular mundial	11
2.2. Venta vehicular mundial	12
2.3. Producción vehicular en México	13
2.4. Venta vehicular en México	14
2.5. Exportación vehicular de México	14
2.6. Esquema del proceso KDD	17
2.7. Gráfica de líneas con varias series de datos	21
2.8. Diagrama de caja y bigotes	22
2.9. Esquema de una neurona artificial	29
2.10 Comparación entre funciones de activación	32
2.11 Diagrama de Perceptrón multicapa MLP	35
2.12 Proceso de convolución con un filtro deslizante.	38
2.13 Ejemplo del proceso de max pooling	39
2.14 Modelo esquemático de una RNN	40
2.15 Modelo esquemático de una celda LSTM	42
2.16 Modelo esquemático de la GRU	44
2.17 Comparación entre búsqueda por malla y aleatoria	48
3.1. Diagrama de flujo del método	54
3.2. Datos de ventas	56
3.3. Archivo de ventas preparado para su uso en el modelo de predicción . .	59
3.4. Lectura de archivo CSV en Power BI	76

4.1. Comparación entre ventas reales y predicción	84
4.2. Comportamiento de venta por número de parte	85
4.3. Resultado de búsqueda por malla	87
4.4. Ejemplo del reemplazo de valores atípicos	89
4.5. Comparación de datos originales y valores atípicos fijados.	89
4.6. Diagrama de caja de un producto con venta semanal	91
4.7. Predicción del modelo GRU en pandemia	93
4.8. Comparación de datos en pandemia	95
4.9. Generación de scrap y producción	99
4.10 Generación mensual de desperdicio en 2023	100
4.11 Predicción de producción total con desperdicio	101

Índice de tablas

2.1. Comparación entre RNN, LSTM y GRU	47
4.1. Resultados obtenidos del modelo GRU	83
4.2. Números de parte con mayor cantidad de venta	85
4.3. Resultado de métricas del modelo GRU	88
4.4. Resultados del modelo GRU con valores atípicos fijados al máximo anual.	90
4.5. Resultados del modelo GRU con conjunto de datos original	92
4.6. Predicción del modelo GRU para el año 2023 (original)	96
4.7. Predicción del modelo GRU para el año 2023 (sustituido)	96
4.8. Predicción del modelo GRU para el año 2023 (desde 2021)	97
4.9. Predicción del modelo GRU con scrap	101

Capítulo 1

Introducción

Para el desarrollo de la presente tesis, se utilizan datos proporcionados por una empresa privada del sector automotriz, especializada en la manufactura de sensores de presión destinados a vehículos. El proyecto se estructura en tres fases principales, cada uno con objetivos y enfoques específicos.

La primer atapa se enfoca en la generación de un modelo de prognosis de ventas, el cual se construye a partir del análisis de datos históricos suministrados por la empresa. Para llevar a cabo esta tarea, se implementa una red neuronal recurrente, particularmente utilizando una arquitectura basada en unidades recurrentes cerradas (*Gated Recurrent Unit, GRU*). Este tipo de red es especialmente adecuado para la modelación de series temporales debido a su capacidad para capturar dependencias a largo plazo en los datos secuenciales. La predicción de ventas resultante de este modelo servirá como base para proyectar el volumen de piezas a fabricar.

Posteriormente, se integra esta estimación con información adicional relacionada con las tasas de generación de desperdicio conocido en el contexto de manufactura como scrap que se producen durante el proceso de producción. Esta consideración permite realizar ajustes más precisos a la planeación de la producción, al tener en cuenta las pérdidas inevitables en el sistema.

Finalmente, con base en las predicciones de ventas, las tasas de scrap y la capacidad instalada de producción, se desarrolla un sistema de apoyo a la toma de decisiones

orientado a la planeación eficiente de la manufactura. Este sistema permite proponer esquemas de producción que optimicen los recursos disponibles, minimicen el desperdicio y satisfagan la demanda proyectada de forma oportuna.

1.1. Antecedentes

Aunque la Ciencia de Datos comenzó a consolidarse en la década de 1990 en Estados Unidos, como resultado de la convergencia entre diversas disciplinas entre ellas las matemáticas, la programación y el análisis de negocios, es apenas en los últimos años que ha empezado a cobrar una relevancia significativa en el contexto mexicano. Esta adopción tardía ha generado una necesidad urgente de incorporar dichas tecnologías en distintos sectores industriales, especialmente en aquellos donde la competitividad y la eficiencia operativa resultan cruciales para el éxito organizacional.

México se distingue como una economía orientada a la manufactura, caracterizada por su papel como plataforma de producción para la exportación. Debido a su ubicación geográfica estratégica, acuerdos comerciales internacionales y mano de obra especializada, numerosas empresas extranjeras han establecido plantas productivas en territorio mexicano, aprovechando el país como centro de transformación de insumos y ensamblaje de productos terminados para los mercados globales.

En particular, la industria automotriz es uno de los pilares de la economía nacional. Según lo reportado por la Asociación Mexicana de la Industria Automotriz (AMIA) ([AMIA, 2023](#)), tras el impacto económico derivado de la pandemia de COVID-19 y la crisis global de semiconductores, la producción automotriz mundial experimentó una disminución del 15.8% entre 2019 y 2020. Aunque en 2021 se observó un modesto crecimiento del 3.1% respecto al año anterior, esta cifra aún se encontraba lejos de los niveles pre-pandémicos. A estas dificultades se sumaron otros factores adversos como el aumento de los costos de financiamiento y una inflación elevada, lo que afectó directamente la demanda de vehículos. Si bien para febrero de 2023 el panorama comenzó a mejorar con una mayor estabilidad en las cadenas de suministro, lo cual permitió una mayor disponibilidad de unidades nuevas, las cifras de ventas aún no

lograban alcanzar las expectativas del sector.

Un fenómeno adicional que ha influido en la dinámica del mercado nacional es la entrada de marcas automotrices de origen chino, que intensificaron la competencia al ofrecer modelos accesibles y bien equipados. Esta incursión ha implicado una pérdida de participación de mercado para algunas marcas ya establecidas en México, generando una presión adicional para mejorar la eficiencia y competitividad de las empresas locales y extranjeras asentadas en el país.

En este contexto desafiante, la industria de autopartes ha comenzado a adoptar estrategias basadas en tecnologías emergentes para incrementar su capacidad de respuesta ante las exigencias del mercado. Entre estas estrategias, destaca la implementación de herramientas basadas en Ciencia de Datos e Inteligencia Artificial (IA), orientadas a fortalecer la toma de decisiones empresariales mediante el análisis automatizado de grandes volúmenes de información.

La predicción y planeación de la producción constituyen áreas clave en las que la IA ha demostrado aportar ventajas competitivas. Diversas compañías de sectores variados están integrando modelos de aprendizaje automático (*machine learning*) en sus procesos, lo que ha elevado el nivel de exigencia en la industria. En consecuencia, aquellas organizaciones que no logren adoptar e integrar estas herramientas de manera efectiva podrían enfrentar serias dificultades para mantenerse en el mercado, o incluso quedar rezagadas frente a sus competidores.

Motivada por esta realidad, la presente tesis se enfoca en el estudio y desarrollo de soluciones basadas en aprendizaje automático aplicadas al proceso de manufactura de sensores automotrices. En particular, se analizan y diseñan modelos que permitan predecir la demanda de ventas a partir de datos históricos, lo cual representa el primer paso para una planeación de la producción más precisa. Posteriormente, se aborda el desafío de estimar la producción real, considerando tanto el tiempo efectivo de operación como la pérdida generada por piezas que no cumplen con los estándares de calidad, comúnmente denominadas *scrap*.

Integrando estos elementos, se busca desarrollar un sistema automatizado de planeación de la producción, capaz de brindar recomendaciones óptimas considerando

tanto la demanda estimada como las restricciones operativas del proceso. El objetivo final es contribuir a la optimización de la cadena de suministro, reducir desperdicios y mejorar la eficiencia global de la manufactura de sensores de presión automotrices en un entorno altamente competitivo y cambiante.

1.2. Planteamiento del Problema

1.2.1. Definición del problema

Para que una empresa logre ser rentable se debe abordar inicialmente un concepto que es primordial para las sanas finanzas, el EBIT (Earnings Before Interests and Taxes, por sus siglas en inglés), que son las ganancias antes del pago de impuestos, y este número nos indica, si es positivo, que la empresa está creciendo. Existen dos maneras de poder incrementar este indicador: aumentando los precios, lo que podría afectar las relaciones con los clientes, además de incrementar el precio de los vehículos al consumidor final; o bien, disminuir costos y seguir ofreciendo excelentes productos a un precio que los consumidores quieran y puedan adquirir.

Las ventas es otro indicador fundamental en cualquier negocio, y para ello se hacen proyecciones hacia el futuro. Lograr saber la cantidad de unidades que se van a vender es crucial para la planeación correcta de recursos en diferentes áreas como lo son, logística, producción, recursos humanos y finanzas. No obstante, actualmente no se cuenta con algún proceso estadístico estándar para pronosticar las ventas de forma automática en esta empresa Alemana. A su vez, para lograr cumplir con la demanda se escogen las líneas de producción con base a la disponibilidad, es decir, no se tiene un sistema que realice la gestión automática y que evalúe dónde es mejor fabricar algún producto. Es por ello que, se requiere un sistema que no solamente tome en cuenta la demanda de ventas, sino que, evalúe en cuál línea de producción se generan menos desperdicios, disminuyendo las tareas manuales, logrando así, flexibilidad ante los cambios repentinos y un EBIT positivo sin incrementar los precios.

Implementar un sistema basado en inteligencia artificial que pueda optimizar la

producción tomando como base el pronóstico generado por modelos estadísticos y que además reconozca la correlación entre producción y desperdicio generaría un punto de inflexión demostrando que es viable el uso de herramientas basadas en inteligencia artificial enfocadas a la manufactura. ¿Es posible planear la producción de sensores de presión automotrices con un sistema inteligente que tome en cuenta las correlaciones entre la producción con el desperdicio; y que además asegure el cumplimiento de la demanda con los clientes?

1.3. Justificación

El sector automotriz en México, particularmente en el ámbito de la manufactura, aún presenta una adopción limitada de herramientas basadas en ciencia de datos y aprendizaje automático. Si bien estas tecnologías han demostrado su eficacia en diversos sectores a nivel global, su aplicación en los procesos de producción sigue siendo incipiente en muchas plantas mexicanas. La incorporación de estas herramientas tiene como objetivo fundamental mejorar la eficiencia operativa en el día a día de las organizaciones, reduciendo los tiempos de retraso en las entregas, optimizando tareas repetitivas y tediosas, minimizando errores humanos y, al mismo tiempo, brindando soporte a los trabajadores para que puedan enfocar su tiempo en actividades de mayor valor agregado.

A nivel nacional, el sector bancario se destaca como el más avanzado en la adopción de tecnologías orientadas al análisis y explotación de datos, seguido por sectores como el militar, el aeronáutico y, en menor medida, el manufacturero. En el extremo opuesto se encuentran sectores como el agrícola y aquellos relacionados con la extracción y procesamiento de materias primas, donde el uso de estas tecnologías es todavía muy limitado. Esta disparidad evidencia una brecha importante en términos de digitalización y modernización tecnológica. La presente investigación se centra específicamente en el área manufacturera, con énfasis en el subsector de autopartes del sector automotriz. A través del desarrollo de esta tesis, se busca contribuir a la incorporación de herramientas tecnológicas avanzadas en este campo, ayudando

así a cerrar la brecha existente con respecto a sectores más desarrollados como el financiero.

En la actualidad, la planeación de la producción en muchas empresas del sector se realiza con base en el análisis manual de la demanda del cliente y la disponibilidad operativa de las líneas de producción. Este proceso depende en gran medida de la experiencia y criterio del personal especializado en logística de producción, quienes, en conjunto con el área de manufactura, determinan la asignación de recursos y la programación de las actividades productivas. Este tipo de planeación se lleva a cabo diariamente y puede requerir entre dos y cuatro horas semanales de trabajo, involucrando al menos a dos personas. Sin embargo, este procedimiento presenta limitaciones importantes, ya que no contempla factores críticos como el desperdicio de material generado durante el proceso (*scrap*). Como consecuencia, no es raro que el plan de producción no se cumpla conforme a lo previsto, lo que puede derivar en retrasos y afectar el cumplimiento de entregas a los clientes.

La utilización de datos históricos y en tiempo real como base para la toma de decisiones representa una estrategia prometedora para mejorar la precisión en la estimación de la demanda y, por consiguiente, optimizar la planeación de la producción. Al integrar estas predicciones con estadísticas del proceso productivo como tasas de desperdicio, capacidades operativas y tiempos de ciclo es posible automatizar parte del proceso de planeación, reducir la carga operativa del personal encargado, atender con mayor rapidez situaciones críticas y, en general, elevar la calidad y confiabilidad de las decisiones operativas.

Existe una percepción generalizada de que la introducción de tecnologías basadas en inteligencia artificial y automatización podría desplazar a los trabajadores que actualmente desempeñan estas labores. No obstante, esta visión está alejada de la realidad. Lejos de sustituir al personal, estas herramientas están diseñadas para complementar sus capacidades, facilitando su trabajo y permitiéndoles enfocarse en actividades más estratégicas y menos repetitivas. Lo que se requiere es una actitud proactiva frente al cambio, acompañada del desarrollo de habilidades para adaptarse al uso de estas nuevas tecnologías. La transición hacia entornos productivos más

automatizados y basados en datos no implica la eliminación de empleos, sino una transformación de los roles laborales, donde el conocimiento técnico y la capacidad de interpretación de información se vuelven cada vez más relevantes.

1.4. Objetivos

1.4.1. Objetivo general

Sistematizar la planeación de la producción de sensores automotrices de presión para el año 2024, considerando las estadísticas de producción y ventas del periodo 2018 a 2023, cumpliendo con la predicción de demanda de los clientes.

1.4.2. Objetivos específicos

- Determinar el modelo de predicción de ventas mediante el análisis de series temporales con el uso de redes neuronales recurrentes.
- Indicar las reglas de asociación existentes entre las actividades de producción y ventas, mediante estadística descriptiva.
- Definir el sistema de recomendación que evalúe las mejores líneas para producir los sensores automotrices de presión con inteligencia artificial.
- Realizar la manipulación, selección y procesamiento de los datos con el lenguaje de programación python.

1.5. Hipótesis

Se podrá predecir la producción de sensores (planeación) utilizando herramientas de aprendizaje automático; como redes neuronales recurrentes para evaluar la predicción de las ventas, lo anterior, considerando como factores la demanda de ventas y material desperdiciado para cumplir con el tiempo estipulado con la entrega de producción con los clientes.

Capítulo 2

Fundamentos

2.1. Marco teórico

2.1.1. Rentabilidad de las empresas

La rentabilidad de una empresa es un indicador fundamental que permite evaluar su viabilidad económica y su capacidad para mantenerse operativa en el largo plazo. Para que un negocio sea considerado rentable, debe ser capaz de cubrir de manera eficiente los costos asociados a su operación, cumplir con sus obligaciones fiscales incluidos los pagos de impuestos, y además generar un excedente que represente utilidades para sus propietarios o accionistas.

En términos financieros, esto implica que los ingresos generados por la empresa deben ser consistentemente superiores a sus egresos. Una forma común de medir esta condición es mediante el análisis del beneficio antes de intereses e impuestos (*EBIT*, por sus siglas en inglés), el cual proporciona una visión clara del rendimiento operativo sin considerar la estructura de financiamiento o los efectos fiscales. Un *EBIT* positivo indica que la empresa está generando suficientes ingresos operativos para cubrir sus costos fijos y variables, lo cual es un signo de salud financiera.

La rentabilidad, por tanto, no solo implica la generación de ganancias, sino también la eficiencia con la que una organización utiliza sus recursos para producir valor

económico. Este aspecto se vuelve especialmente relevante en sectores como el manufacturero, donde la optimización de procesos, la reducción de desperdicios y una adecuada planeación de la producción pueden incidir directamente en los márgenes de rentabilidad.

Ingresos económicos de las empresas

Los ingresos económicos de una empresa provienen, en términos generales, de la venta de productos o servicios relacionados con su objeto social o giro comercial. Estos ingresos constituyen la fuente primaria de entrada de recursos financieros, y su volumen y constancia son determinantes para la sostenibilidad del negocio. En el contexto manufacturero, como es el caso del sector de autopartes, los ingresos suelen estar directamente vinculados al cumplimiento de pedidos y contratos establecidos con clientes nacionales e internacionales.

Egresos económicos de las empresas

Los egresos económicos se refieren a todos aquellos desembolsos que realiza la empresa para llevar a cabo sus operaciones. Estos gastos se dividen, comúnmente, en dos categorías principales: gastos fijos y gastos variables.

Los gastos fijos son aquellos que se mantienen relativamente constantes, independientemente del volumen de producción o nivel de ventas. Ejemplos típicos de este tipo de egresos incluyen los costos asociados a áreas de soporte como recursos humanos, tecnologías de la información, seguros, arrendamientos y depreciación o devaluación de maquinaria.

Por otro lado, los gastos variables están directamente relacionados con el nivel de actividad productiva. Estos aumentan o disminuyen en proporción a la cantidad de productos fabricados o servicios prestados. Dentro de esta categoría se encuentran los costos de materiales directos, salarios del personal operativo en planta, consumo de energía, costos logísticos y otros insumos necesarios para la producción.

Ganancias antes de impuestos (*EBIT*)

El indicador conocido como *EBIT* (por sus siglas en inglés, Earnings Before Interest and Taxes) representa la utilidad operativa que una empresa obtiene antes de deducir los intereses financieros y las obligaciones fiscales. Se considera uno de los parámetros más relevantes para evaluar la eficiencia operativa y la rentabilidad de una organización, ya que permite identificar si el negocio genera suficientes recursos con sus actividades principales, sin verse afectado por la estructura de financiamiento o las políticas impositivas.

El *EBIT* se calcula mediante la siguiente expresión:

$$EBIT = Ingresos - Egresos \quad \text{(Ecuación 2.1)}$$

Obtener un *EBIT* positivo implica que la empresa no solo está cubriendo sus costos de operación, sino también generando utilidades operativas. Para alcanzar esta condición, existen principalmente dos estrategias: incrementar los ingresos mediante una mayor eficiencia comercial o ampliación del mercado, y reducir los egresos a través de la optimización de procesos, control de gastos y mejora en el aprovechamiento de recursos.

2.1.2. Mercado vehicular y de auto partes

El análisis del mercado vehicular resulta fundamental para comprender el comportamiento y las tendencias en la industria de autopartes. Dado que esta última depende directamente de la producción y comercialización de vehículos, el estudio de los indicadores relevantes permite establecer proyecciones y tomar decisiones estratégicas con mayor precisión. En esta sección se examinan dos variables clave: la producción y las ventas de vehículos, con base en los datos reportados por la Asociación Mexicana de la Industria Automotriz (AMIA) en el periodo comprendido entre 2010 y 2023.

Mercado vehicular y de auto partes en el mundo

A nivel global, la industria automotriz ha experimentado una evolución constante durante las últimas décadas. Como se muestra en la Figura 2.1, la producción mundial de vehículos presentaba una tendencia creciente sostenida hasta el año 2019. Sin embargo, en 2020 se observó una caída significativa como consecuencia directa de la crisis sanitaria provocada por la pandemia de COVID-19. Esta contingencia no solo afectó las operaciones de manufactura y las cadenas de suministro a nivel internacional, sino que también redujo la demanda de vehículos debido al confinamiento y la incertidumbre económica.

El comportamiento de la producción vehicular influye directamente en el mercado de autopartes, ya que cada unidad fabricada requiere componentes especializados desde sensores hasta sistemas completos que deben ser abastecidos por empresas del sector. Por tanto, cualquier variación significativa en la producción vehicular tiene un impacto directo en la demanda de autopartes, lo que justifica la necesidad de estudiar estas dinámicas como parte del diagnóstico estratégico de la industria.

Producción mundial de vehículos (unidades)

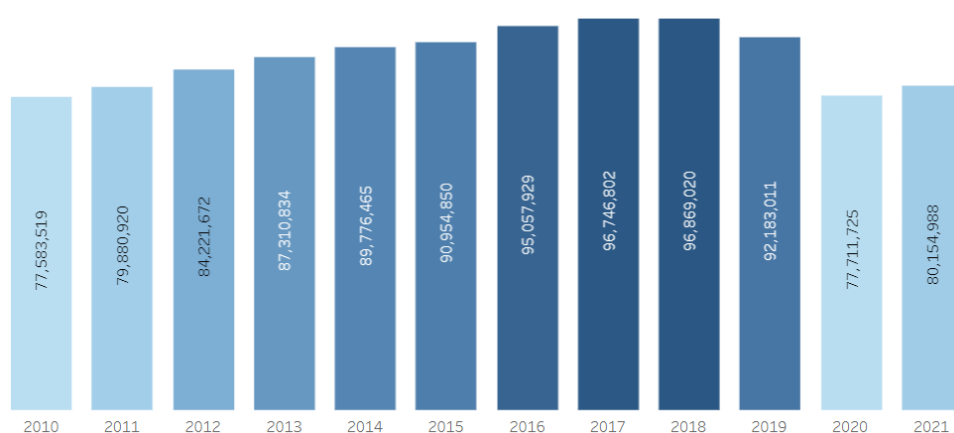


Figura 2.1: Producción vehicular mundial. Imagen obtenida de AMIA

Asimismo, en la Figura 2.2 se observa que las ventas globales de vehículos siguen una tendencia similar a la de la producción. En particular, durante el año 2021, los niveles de venta registrados se asemejan a los observados casi una década atrás, evidenciando una significativa contracción en el mercado.

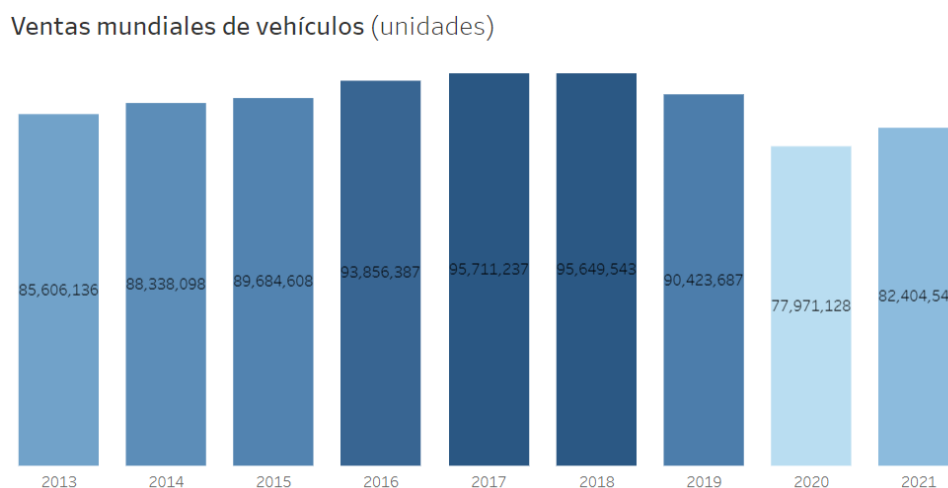


Figura 2.2: Venta vehicular mundial. Imagen obtenida de AMIA

A nivel mundial, los efectos de la pandemia por COVID-19 resultaron profundamente disruptivos para la industria automotriz. A pesar de los esfuerzos por reactivar la economía y estabilizar las cadenas de suministro, los volúmenes de venta aún no han logrado alcanzar los niveles previos a la contingencia sanitaria. Esta situación pone de manifiesto la magnitud de los impactos sufridos y subraya la importancia de adoptar estrategias apoyadas en tecnologías emergentes, para enfrentar futuros escenarios de incertidumbre.

Este retroceso evidenció la vulnerabilidad del sector ante eventos disruptivos globales, así como la necesidad de adaptar los modelos de negocio hacia esquemas más resilientes y basados en datos. La reactivación posterior ha sido gradual y heterogénea entre países, condicionada por factores como la disponibilidad de semiconductores, políticas gubernamentales de estímulo económico, y cambios en las preferencias del consumidor.

Mercado vehicular y de auto partes en México

En el caso de México, el Instituto Nacional de Estadística y Geografía (INEGI) proporciona datos mensuales sobre la producción, venta y exportación de vehículos ligeros, lo que permite un análisis detallado de la evolución del sector automotriz a lo largo del tiempo.

Tal como se observa en la Figura 2.3, la producción de vehículos en el país sufrió una caída drástica en marzo de 2020, alcanzando niveles cercanos a cero como consecuencia de la suspensión de actividades derivada de la pandemia por COVID-19. No obstante, esta situación fue temporal, y para el año 2023, los niveles de producción lograron recuperarse y alcanzar cifras similares a las observadas antes de la pandemia. Cabe destacar que esta recuperación no se debió a un aumento en las ventas internas, ya que, como se muestra en la Figura 2.4, el mercado nacional no ha logrado retornar a los niveles de comercialización previos a 2020. En cambio, la principal razón detrás del repunte en la producción es el crecimiento sostenido de las exportaciones, como se aprecia en la Figura 2.5.

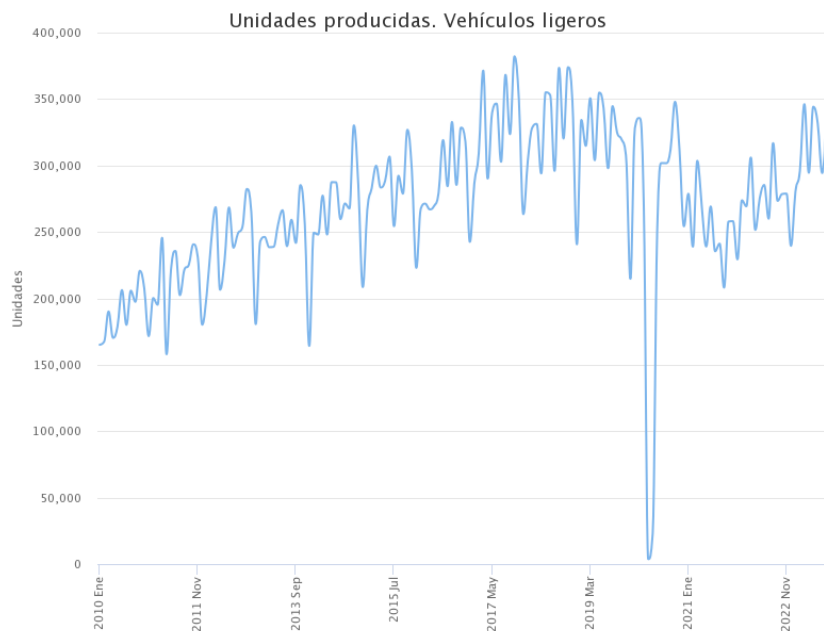


Figura 2.3: Producción vehicular en México. Imagen obtenida de INEGI

Estas gráficas evidencian que el mercado automotriz mexicano ha comenzado a retomar la tendencia alcista que presentaba antes del 2020, especialmente en lo que respecta a la producción destinada al mercado internacional. Sin embargo, el estancamiento en las ventas internas sugiere que la recuperación del sector aún no ha sido total, situación que persiste incluso hacia el año 2024.

Este tipo de análisis es esencial para comprender la dinámica actual de la industria

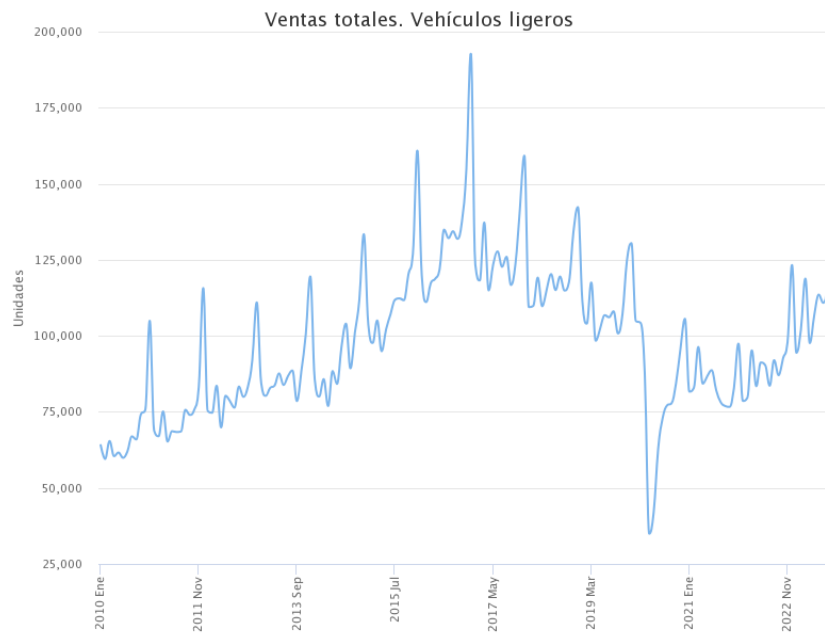


Figura 2.4: Venta vehicular en México. Imagen obtenida de INEGI

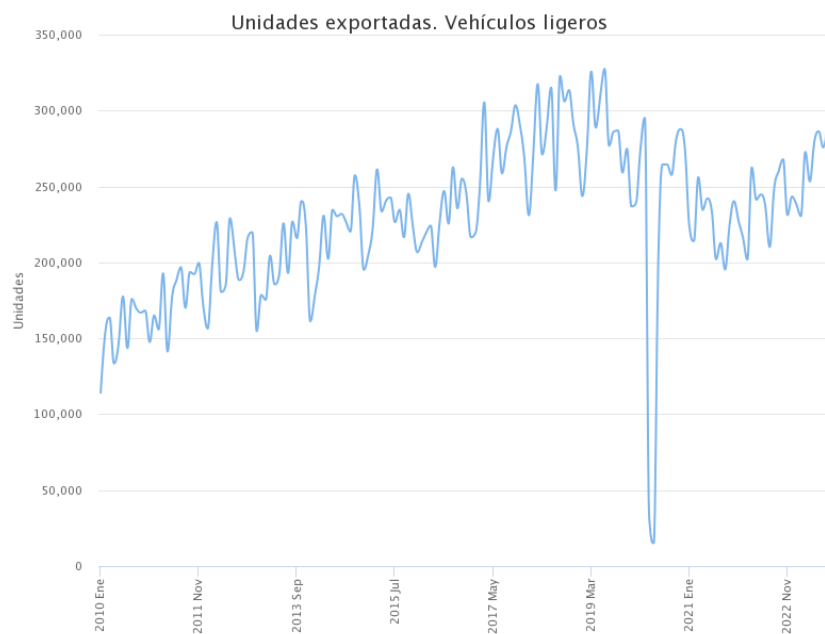


Figura 2.5: Exportación vehicular de México. Imagen obtenida de INEGI

automotriz nacional, así como para contextualizar el desarrollo e implementación de soluciones tecnológicas, como los sistemas de predicción y planeación de la producción, los cuales pueden contribuir a una mayor eficiencia en el contexto de una recuperación aún parcial del mercado interno.

2.1.3. Proceso de cadena de valor en los sensores de presión automotrices

El proceso de manufactura de sensores de presión automotrices involucra múltiples etapas interconectadas, que abarcan desde la adquisición de materia prima hasta la entrega del producto final al cliente. Este flujo de valor puede dividirse en tres bloques principales: obtención de insumos, transformación (manufactura), y logística de distribución.

Obtención de materia prima

La primera etapa consiste en la adquisición de los materiales necesarios para la producción, los cuales son suministrados por distintos proveedores nacionales e internacionales. Esta fase implica un análisis logístico y de gestión de compras, considerando variables como tiempos de entrega, disponibilidad, calidad del material y costo-beneficio, con el fin de asegurar el abastecimiento oportuno para la continuidad de la operación.

Elaboración de productos

La planeación de la producción es realizada por un especialista en logística de producción, quien determina las cantidades requeridas de producto y asigna dichas cantidades a las líneas específicas donde serán fabricadas. Generalmente, esta planificación se ejecuta considerando el 80% de la capacidad total de cada línea. Por ejemplo, si una línea tiene la capacidad de producir 100,000 unidades en un periodo de 24 horas, la planificación se realiza sobre la base de 80,000 unidades para ese mismo periodo.

El 20% restante de la capacidad no se considera explícitamente dentro del plan de producción, ya que se asume que será absorbido por factores no modelados con precisión, tales como paros de línea ocasionados por mantenimiento no programado, así como por la generación de piezas defectuosas (desperdicio o *scrap*). Estos elementos, aunque previsibles, no suelen ser cuantificados con exactitud en la etapa de planeación, lo cual genera incertidumbre en la ejecución real del plan.

Entrega de productos a clientes

Una vez finalizada la producción, los productos terminados son despachados hacia los clientes. Esta etapa vuelve a formar parte del área de logística, y generalmente se lleva a cabo mediante transporte terrestre (camiones) o marítimo, dependiendo del destino. Después del envío, se estima que la facturación correspondiente se efectúe en un plazo promedio de tres días.

No obstante, debido a los alcances delimitados del presente proyecto y a restricciones de tiempo, las variables asociadas al proceso de distribución y entrega no serán consideradas dentro del desarrollo de esta investigación.

Cabe señalar que todos los datos operativos utilizados en esta investigación fueron extraídos del sistema de gestión empresarial SAP, una plataforma de origen alemán ampliamente utilizada en entornos industriales para la planificación y control de recursos empresariales ([SAP, 2024](#)). El acceso a los datos está restringido por roles y niveles de autorización definidos por la empresa. Para su análisis, los datos fueron exportados en formato .x/sx, lo que permitió su manipulación y procesamiento con herramientas externas de análisis de datos.

2.1.4. Descubrimiento de conocimiento en bases de datos (KDD)

Como señala ([Hernandez, 2004](#)), el proceso de Descubrimiento de Conocimiento en Bases de Datos (*KDD*, por sus siglas en inglés) se caracteriza por ser iterativo y no lineal. Esta característica implica que los resultados obtenidos en una etapa pueden requerir

el retorno a fases previas, ya sea para ajustar parámetros, redefinir objetivos o mejorar la calidad del análisis. El propósito principal del KDD es la generación de conocimiento útil y comprensible a partir de grandes volúmenes de datos. La estructura general de este proceso se muestra en la Figura 2.6.

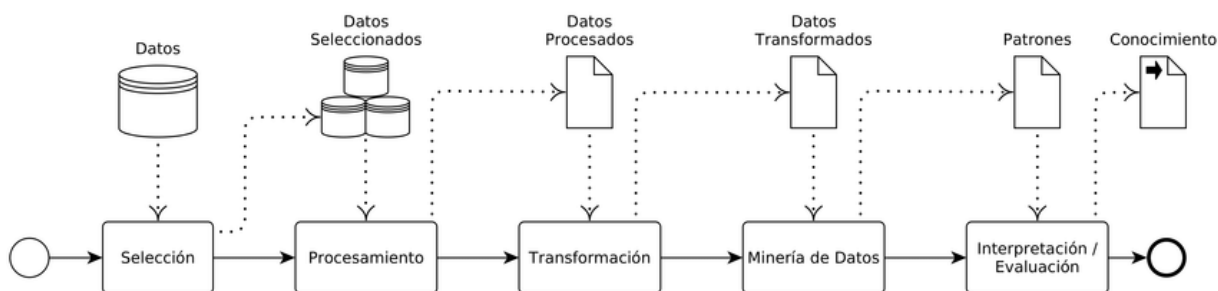


Figura 2.6: Esquema del proceso KDD. Fuente: (Rottoli, 2018)

Selección

Los datos provienen de fuentes de información que son veraces, en esta fase se realiza la integración y la recopilación de datos que serán útiles para el conocimiento que se pretende obtener.

En esta primera fase del proceso de *KDD*, se identifican y extraen los datos relevantes desde diversas fuentes confiables de información. En esta fase se realiza la integración y la recopilación de aquellos conjuntos de datos que tengan una relación directa con el problema de análisis, de modo que su procesamiento posterior pueda conducir a la obtención de conocimiento útil.

Procesamiento

La fase de procesamiento, también conocida en la literatura como análisis exploratorio de datos (EDA, por sus siglas en inglés) o estadística descriptiva, tiene como propósito fundamental comprender la estructura y características básicas del conjunto de datos seleccionado. Esta etapa es crucial para evaluar la calidad de los datos y preparar adecuadamente su transformación y análisis posterior.

Durante esta fase, se realiza la identificación de los tipos de datos presentes (numéricos, categóricos, temporales, etc.), la detección de datos faltantes, el reconocimiento de valores atípicos (outliers) y la identificación de registros erróneos o inconsistentes que se desvían de los patrones esperados. Estas tareas permiten conocer el estado real del conjunto de datos y determinar la necesidad de aplicar técnicas de limpieza, imputación o normalización.

Esta etapa da lugar al conjunto de datos, representado como $D = \{(X_i, \mathbf{y}_i)\}_{i=1}^m$, donde $X \in \mathbb{R}^N$ corresponde al conjunto de atributos o características y N la cantidad de atributos, $\mathbf{y} \in \mathbb{R}^{|y|}$ representa la variable objetivo que se desea predecir o clasificar y cuya predicción se representa como $\hat{\mathbf{y}}$ donde $|y|$ es el dominio de $\mathbf{y} \in \mathbb{R}^{|y|}$ que puede ser un dato numérico o categórico para $|y| \geq 2$ y la cantidad de registros es determinada como m (Liu, 2006). Esta representación es fundamental, ya que define la vista de los datos que el modelo utilizará para aprender patrones y relaciones internas. .

Como refieren (Alabi y Bukola, 2023), la estadística descriptiva brinda información valiosa de los datos como variabilidad, tendencias y distribución. Con ello se logra sumarizar la información, y detectar errores u *outliers* (los *outliers* sesgan los análisis). De esta manera se pueden encontrar patrones y correlaciones en los datos.

Asimismo, se emplean medidas estadísticas como la media, mediana, moda, desviación estándar, entre otras, que permiten describir el comportamiento general de las variables. Además, se utilizan representaciones gráficas como histogramas, diagramas de caja (*boxplots*), gráficos de dispersión y mapas de calor para visualizar posibles patrones, correlaciones o irregularidades.

Las métricas más comunes de la estadística descriptiva, que se pueden encontrar en (Bruce, 2020), son:

Promedio: Se calcula sumando todos los valores dividiéndolo entre el número total valores.

$$\text{Promedio} = \bar{y} = \frac{\sum_{i=1}^m y_i}{m} \quad (\text{Ecuación 2.2})$$

donde m es la cantidad de valores, y_i es la variable objetivo numérica en la posición i .

Mediana: La mediana es el valor central de una lista ordenada de números. Esta métrica es más robusta que el promedio cuando existen *outliers*.

La mediana para datos de longitud impar es el valor medio.

$$\text{Mediana impar} = y_{medio} \quad (\text{Ecuación 2.3})$$

La mediana para datos de longitud par es el promedio del valor medio inferior con el medio superior.

$$\text{Mediana par} = \frac{y_{medio-1} + y_{medio+1}}{2} \quad (\text{Ecuación 2.4})$$

Moda: La moda es el valor que más veces se repite.

$$\text{Moda} = c_{max}^{(y)} \quad (\text{Ecuación 2.5})$$

donde c cuenta las repeticiones de los datos en y

Rango: La diferencia entre el valor más grande y el más pequeño. Este se ve afectado en presencia de *outliers*.

$$\text{Rango} = y_{max} - y_{min} \quad (\text{Ecuación 2.6})$$

Varianza: La sumatoria de las desviaciones cuadradas al promedio dividida por $m - 1$ donde m es el número de datos.

$$\text{Varianza} = s^2 = \frac{\sum_{i=1}^m (y_i - \bar{y})^2}{m - 1} \quad (\text{Ecuación 2.7})$$

Desviación estándar: La raíz cuadrada de la varianza. Regularmente se utiliza para interpretar datos. Pero computacionalmente es más difícil de procesar.

$$\text{Desviación estándar} = s = \sqrt{\text{Varianza}} = \sqrt{s^2} \quad (\text{Ecuación 2.8})$$

Coefficiente de correlación de Pearson: Este número estima la relación que existe entre dos variables (y_a e y_b) que se sitúan en la misma escala. Este coeficiente no es robusto en presencia de *outliers*. Si $r = 1$ existe una correlación perfecta positiva, en $r = 0$ no hay correlación y en $r = -1$ hay una correlación perfecta negativa.

$$r = \frac{\sum_{i=1}^m (y_{ai} - \bar{y}_a)(y_{bi} - \bar{y}_b)}{(m-1)s_a s_b} \quad (\text{Ecuación 2.9})$$

En caso de existir *outliers* se pueden usar los coeficientes de Spearman (ρ) o de Kendall (τ) ya que estos utilizan el rango. (Dong, 2023)

Coefficiente de determinación (R^2): Mide la fracción de la variación total en la variable dependiente que es capturada por el modelo de regresión. Con un $R^2 = 0$ nuestro modelo de regresión es aquel que predice solo el promedio de los datos, y no lo hace mejor que ello. Mientras más cercano es nuestro R^2 a 1, es mejor el modelo.

$$R^2 = 1 - \frac{\sum_{i=1}^m (y_i - \hat{y}_i)^2}{\sum_{i=1}^m (y_i - \bar{y})^2} \quad (\text{Ecuación 2.10})$$

donde $\bar{y}$ es el promedio de los valores y_i , ver Ecuación 2.2. $\hat{y}_i$ son los valores predichos por el modelo.

Cociente anualizado de lo real contra la predicción (YoY): Esta evaluación nos permite saber la relación de las ventas totales anuales predichas contra las reales tomando en cuenta todo el periodo de predicción.

$$YoY = \frac{\sum_{i=1}^m \hat{y}_i}{\sum_{i=1}^m y_i} \quad (\text{Ecuación 2.11})$$

donde $\bar{y}$ es el promedio de los valores y_i , ver Ecuación 2.2. $\hat{y}_i$ son los valores predichos por el modelo.

Las métricas estadísticas previamente mencionadas, junto con el uso de visualizaciones gráficas, desempeñan un papel fundamental en la detección de patrones, tendencias y comportamientos anómalos en los datos. Este análisis preliminar permi-

te obtener conocimiento *a priori*, es decir, una comprensión inicial que no depende aún de modelos predictivos o algoritmos avanzados, pero que resulta crucial para orientar las siguientes etapas del proceso de análisis.

Además, estos recursos se utilizan como apoyo para la validación por parte de expertos en el dominio del problema, quienes pueden interpretar los resultados en función de su conocimiento operativo y experiencia. Asimismo, brindan insumos clave para que los responsables de la toma de decisiones puedan comprender los datos de forma clara, lo que facilita el diseño de estrategias fundamentadas y basadas en evidencia cuantitativa. De esta forma, el análisis exploratorio no solo cumple una función técnica, sino también comunicativa, al hacer accesible la información a diversos perfiles dentro de la organización.

Gráfica de líneas: Como señala (Knafllic, 2015), este tipo de gráficas son las más utilizadas para datos continuos, esto se debe a que cada valor es conectado a otro por medio de una línea. Regularmente en el eje horizontal se colocan intervalos de tiempo y en el eje vertical valores numéricos. Este tipo de gráfica puede mostrar múltiples series de datos. Ver Figura 2.7.

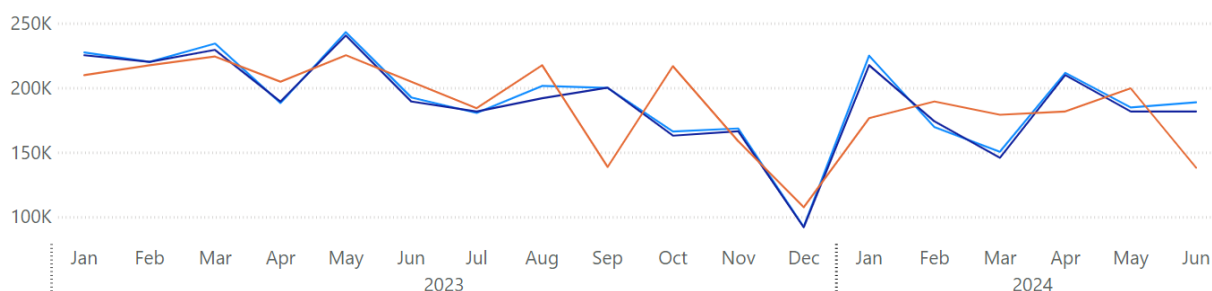


Figura 2.7: Gráfica de líneas con varias series de datos. Fuente: Propio

Diagrama de caja y bigotes (Boxplot): Es una forma rápida de visualizar la distribución de los datos. Fue introducido en los '70's por John Tukey, matemático y estadístico estadounidense. Ver Figura 2.8.

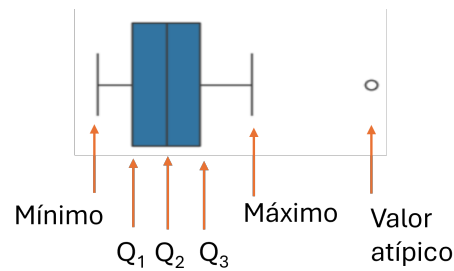


Figura 2.8: Diagrama de caja y bigotes. Fuente: Propio

Transformación

La fase de transformación consiste en preparar y estructurar los datos en una forma adecuada para su análisis mediante técnicas de minería de datos.

Una vez que se tiene la vista de datos minable $D = (X, \mathbf{y})$, idealmente, será dividido en 3 subconjuntos, entrenamiento, validación y pruebas. En la práctica el conjunto de entrenamiento consta del 70-80% de registros, validación el 20% y para pruebas del 0-10%.

El conjunto de entrenamiento está definido por un número de observaciones Q donde $Q \in X$, el de validación por V donde $V \in X$ y el subconjunto de pruebas como P donde $P \in X$.

Existen diferentes maneras de elegir los subconjuntos antes mencionados como lo son por aleatoriedad, porcentaje de cada categoría de y o validación cruzada (k -fold), dependiendo de la naturaleza del problema y del tamaño del conjunto de datos. Es importante señalar que los subconjuntos deben ser excluyentes entre sí, es decir, $Q \cap V \cap P = \emptyset$.

El subconjunto de entrenamiento tiene como propósito permitir que el modelo aprenda los patrones subyacentes en los datos mediante un proceso iterativo de ajuste de parámetros. El conjunto de validación se utiliza para evaluar el comportamiento del modelo durante su desarrollo, especialmente en lo referente a la función de costo, lo que permite aplicar técnicas como el ajuste de hiperparámetros o la detección de sobreajuste (*overfitting*). Finalmente, una vez completado el entrenamiento y la validación, el modelo es evaluado sobre el conjunto de prueba, comparando sus pre-

dicciones $\hat{\mathbf{y}}$ con los valores reales $\mathbf{y}$, con el fin de estimar su rendimiento general sobre datos no vistos previamente.

Dentro de esta fase también se aplican técnicas de normalización o estandarización, cuyo objetivo es llevar todas las variables numéricas a una misma escala. Esto es particularmente importante para modelos sensibles a las magnitudes de los datos. La transformación de datos es fundamental para lograr un correcto modelado. Dentro de este apartado se analizan diferentes tipo de escaladores.

MinMax: Cuando el modelado no requiere un centrado en cero. Con la Ecuación 2.12 el escalador estandariza los valores en el siguiente rango $[0, 1]$. La Ecuación 2.13 realiza un rango $[-1, 1]$.

$$y'_i = \frac{y_i - \min(y)}{\max(y) - \min(y)} \quad (\text{Ecuación 2.12})$$

$$y'_i = \frac{2(y_i - \min(y))}{\max(y) - \min(y)} - 1 \quad (\text{Ecuación 2.13})$$

donde y'_i es el valor normalizado del valor y_i .

Estándar o normal: Este tipo de estandarización se caracteriza por obtener un promedio cero en los datos con un rango $[-1, 1]$. Utiliza el promedio Ecuación 2.2 y la desviación estándar Ecuación 2.8.

$$y'_i = \frac{y_i - \bar{y}}{s_y} \quad (\text{Ecuación 2.14})$$

donde y'_i es el valor normalizado del valor y_i , $\bar{y}$ y s_y son el promedio y desviación estándar de y .

Robusta: Esta normalización permite que la mediana sea cero y mantener intacto al promedio o la varianza en presencia de valores atípicos. El denominador es la

desviación promedio absoluta (MAD)

$$y'_i = \frac{y_i - \text{mediana}(y)}{\text{mad}(y)}$$
$$\text{mad}(y) = \frac{\sum_{i=1}^m |y_i - \text{mediana}(y)|}{m}$$

(Ecuación 2.15)

En resumen, la fase de transformación garantiza que el conjunto de datos esté en una forma adecuada tanto estructural como numéricamente para el proceso de modelado, asegurando así una base sólida para obtener resultados confiables y reproducibles.

Minería de datos

La fase de minería de datos constituye el núcleo del proceso de descubrimiento de conocimiento, ya que en esta etapa se seleccionan, aplican y evalúan los algoritmos que permitirán identificar patrones relevantes, inferencias o relaciones ocultas en el conjunto de datos previamente transformado.

Durante esta fase, se definen tanto los métodos de modelado como las métricas de evaluación que permitirán valorar el desempeño de los modelos construidos. La elección del algoritmo depende en gran medida de la naturaleza del problema a resolver, las características del conjunto de datos y los objetivos específicos del análisis.

En términos generales, las tareas de minería de datos pueden agruparse en dos categorías principales: regresión y clasificación ([Bishop, 2008](#)). La regresión se refiere a problemas en los que la variable objetivo y es continua, y el objetivo es predecir un valor numérico basado en las características X del conjunto de datos. Ejemplos de tareas de regresión incluyen la estimación de ventas, la predicción de temperatura, o el pronóstico de demanda en procesos industriales.

Por otro lado, la clasificación se aplica cuando la variable objetivo es categórica, y el objetivo es asignar nuevas observaciones a una de las clases predefinidas. En este caso, el modelo debe aprender una función de decisión capaz de discriminar entre distintas categorías en función de los atributos observados. Ejemplos típicos de clasificación incluyen el diagnóstico médico (enfermo/no enfermo), la detección de defectos (pieza

aceptada o rechazada) o la segmentación de clientes en estrategias de mercado.

La selección del modelo también está sujeta a la disponibilidad de datos, la complejidad del problema y el costo computacional. Algunos de los algoritmos comúnmente utilizados incluyen la regresión lineal.

La regresión lineal simple es el método más simple para predecir una variable y de acuerdo a otra variable explicativa x . La ecuación que relaciona a estas dos variables está dada por:

$$y = w_1 x + b \quad (\text{Ecuación 2.16})$$

Aquí el sesgo b representa la intersección con el eje de la variable y y w_1 es el coeficiente de la variable explicativa.

Existen otros modelos como regresión logística, máquinas de soporte vectorial (SVM), árboles de decisión, bosques aleatorios (*Random Forest*), redes neuronales profundas (*Deep Learning*) y métodos basados en *boosting* (como *XGBoost* o *LightGBM*).

La correcta definición de las métricas de evaluación es igualmente esencial, ya que permiten cuantificar la eficiencia y la capacidad de generalización del modelo. Entre las métricas más comunes se encuentran el error cuadrático medio (*MSE*) y el coeficiente de determinación (R^2) para problemas de regresión, y la precisión, exactitud, sensibilidad, especificidad, F1-score y el área bajo la curva *ROC* (*AUC-ROC*) en tareas de clasificación.

En síntesis, esta fase es decisiva para transformar los datos preparados en modelos predictivos o clasificatorios útiles para la toma de decisiones, lo que representa el objetivo final del proceso de minería de datos.

Interpretación / Evaluación

Una vez que los modelos han sido aplicados y se han obtenido resultados cuantitativos, la fase de interpretación y evaluación tiene como objetivo analizar estos resultados a la luz del problema planteado. En esta etapa, los expertos en el dominio del conocimiento

ya sea del ámbito técnico, operativo o de negocio interpretan las salidas generadas por los modelos, con el fin de determinar si la solución propuesta es válida, útil y aplicable en el contexto real.

Si durante esta etapa se detecta que el modelo no cumple con los estándares deseados ya sea por bajo desempeño, problemas en los datos, sobreajuste, o falta de interpretabilidad, es común regresar a alguna de las etapas previas del proceso *KDD*, como la transformación de los datos, la selección de atributos, o incluso la redefinición del enfoque de minería. Este carácter iterativo del proceso permite refinar gradualmente el conocimiento generado hasta alcanzar un resultado satisfactorio.

En resumen, esta fase constituye el cierre del ciclo de descubrimiento de conocimiento, y su correcta ejecución asegura que los resultados obtenidos puedan ser integrados de manera efectiva a los procesos de toma de decisiones en el entorno real.

2.1.5. Redes neuronales artificiales y aprendizaje profundo

Las redes neuronales artificiales (*ANN*, por sus siglas en inglés) constituyen una clase de técnicas computacionales que simulan los mecanismos de aprendizaje biológicos ([Aggarwal, 2018](#)). Estas técnicas brindan a las computadoras una manera de aprender a generalizar dentro un entorno o problemática específica. En general existen 3 diferentes tipos de aprendizaje automático los cuales se describen a continuación. ([Raschka, Liu, Mirjalili, y Dzhulgakov, 2022](#)).

Tipos de aprendizaje para redes neuronales

Aprendizaje supervisado: El aprendizaje supervisado, también conocido como aprendizaje inductivo, es un enfoque en el cual los algoritmos de aprendizaje automático son entrenados a partir de ejemplos previamente recolectados y etiquetados, generalmente mediante intervención humana. En este paradigma, el objetivo del modelo es aprender una función que mapee correctamente entradas a salidas, de forma que pueda generalizar este conocimiento a nuevos datos no vistos.

La variable y puede ser de naturaleza numérica en cuyo caso se trata de un problema de regresión o categórica lo cual define un problema de clasificación. Cuando $|y| \geq 2$, se considera que existen múltiples clases posibles, y el modelo debe aprender a asignar correctamente nuevas instancias a una de ellas. La eficacia del aprendizaje supervisado depende tanto de la calidad del conjunto de datos como de la capacidad del modelo para capturar las relaciones subyacentes entre los atributos X y el objetivo y .

Este enfoque es ampliamente utilizado en tareas prácticas como el reconocimiento de imágenes, la detección de fraudes, el diagnóstico médico automatizado, y la predicción de demanda, entre muchas otras aplicaciones.

Aprendizaje por refuerzo: El aprendizaje por refuerzo es un paradigma en el cual un agente autónomo aprende a tomar decisiones óptimas mediante la interacción directa con un entorno. A diferencia del aprendizaje supervisado, en el cual el modelo recibe ejemplos explícitos de entrada y salida, en el aprendizaje por refuerzo no se proporcionan pares de atributos con su correspondiente valor objetivo. En su lugar, el agente recibe una señal de recompensa o castigo como retroalimentación de las acciones que ejecuta, y su objetivo es maximizar la recompensa acumulada a lo largo del tiempo ([Pack Kaelbling y W. Moore, 1996](#)).

El sistema está compuesto por tres elementos principales: el agente, el entorno (ambiente), y una función de recompensa. El agente observa un estado del entorno, toma una acción disponible en dicho estado, y como consecuencia recibe una nueva observación del estado junto con una recompensa. A través de múltiples iteraciones, el agente ajusta su estrategia de decisión con el fin de optimizar su desempeño futuro.

Una forma intuitiva de entender este tipo de aprendizaje es la siguiente:

Ambiente: Estado 65; Acciones posibles 4

Agente: Elijo opción 2

Ambiente: Recibes 7 unidades en recompensa

Agente: No cambio mi criterio

Ambiente: Estado 15; Acciones posibles 2

Agente: Elijo opción 1

Ambiente: Recibes -4 unidades en recompensa

Agente: Cambio mi criterio

:

Aprendizaje no supervisado: El aprendizaje no supervisado representa un enfoque dentro del aprendizaje automático en el cual el conjunto de datos disponible no cuenta con etiquetas o valores objetivos asociados. Formalmente, se puede definir como un conjunto de observaciones $D = (X)$, donde $X \in \mathbb{R}^N$ representa únicamente los atributos o características de los datos, sin una variable de salida y que funcione como guía para el entrenamiento del modelo.

Una de las aplicaciones más comunes en este tipo de aprendizaje es la segmentación o agrupamiento de datos, conocida como *clustering*, cuyo objetivo es identificar subconjuntos o grupos dentro de los datos que compartan características similares. Estos métodos permiten clasificar objetos similares entre sí sin necesidad de una clase predefinida, lo cual resulta útil en casos donde el conocimiento del dominio no permite etiquetar previamente la información.

Además del clustering, otra técnica ampliamente utilizada en el aprendizaje no supervisado es la reducción de la dimensionalidad, que consiste en proyectar los datos a un espacio de menor dimensión, preservando al máximo la información relevante. Esta técnica es especialmente útil cuando se trabaja con conjuntos de datos con un gran número de variables, ya que permite simplificar la representación de los datos, mejorar la eficiencia de los modelos y facilitar su visualización. Entre los métodos más conocidos en esta categoría se encuentran el Análisis de Componentes Principales (PCA) y el t-SNE, ambos aplicables tanto a la compresión de datos como al análisis exploratorio. Esto es de gran ayuda para la compresión de datos y la visualización cuando se tienen múltiples atributos (Liu, 2006).

Elementos básicos de las redes neuronales

Neurona artificial: La neurona artificial es el componente fundamental de las redes neuronales y está inspirada en el comportamiento de las neuronas biológicas, tal como se comprendía en los primeros estudios de neurociencia durante el siglo XX. El primer modelo computacional de una neurona fue propuesto en 1943 por Warren McCulloch y Walter Pitts, marcando el inicio del desarrollo de redes neuronales artificiales.

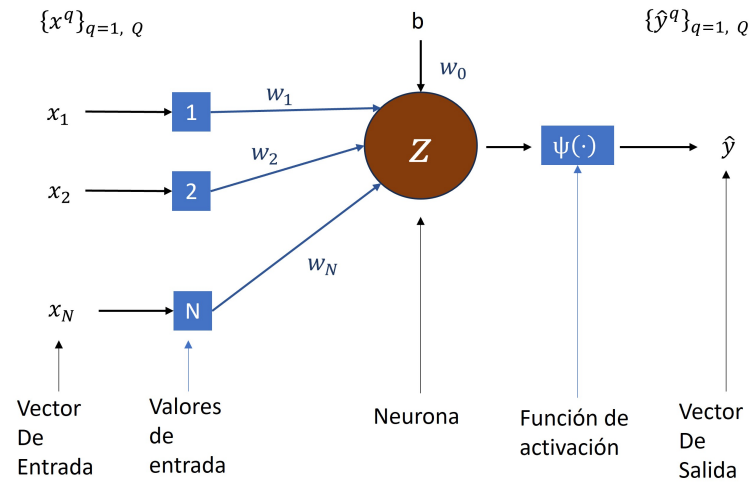


Figura 2.9: Esquema de una neurona artificial. Fuente: Propio

En la Figura 2.9 se muestra el esquema conceptual de este modelo. Desde el punto de vista matemático, una neurona artificial puede representarse como una función que recibe un vector de entradas $\mathbf{x}_N$, donde $\mathbf{x}_N \in Q$, siendo Q el conjunto de datos de entrenamiento. Cada entrada está asociada a un peso correspondiente en el vector $\mathbf{w}_N$, y además se incorpora un término adicional denominado sesgo o bias $\mathbf{b}$, el cual permite ajustar el umbral de activación de la neurona, que pasan por una función de activación $\psi(\cdot)$ y nos da una respuesta $\hat{y}$. También se le conoce como perceptrón.

Estos elementos se combinan linealmente para obtener una señal de activación intermedia z , que se expresa como:

$$z = \sum_{i=1}^N w_i x_i + b \quad (\text{Ecuación 2.17})$$

Si definimos al bias $b = w_0 x_0$ con un valor $x_0 = 1$, la (Ecuación 2.17) se puede modelar a la siguiente (Ecuación 2.18).

$$z = \sum_{i=1}^N w_i x_i + w_0 x_0 = w_0 x_0 + w_1 x_1 + w_2 x_2 + \cdots + w_N x_N = \sum_{i=0}^N w_i x_i \quad (\text{Ecuación 2.18})$$

Posteriormente, el valor de z se introduce en una función de activación $\psi(\cdot)$, la cual tiene como propósito introducir no linealidad al modelo y permitir que la red neuronal aprenda representaciones complejas. El resultado final de la neurona, es decir, su salida o predicción, se denota como $\hat{y}$:

$$\hat{y} = \psi(z) \quad (\text{Ecuación 2.19})$$

Este tipo de estructura básica también recibe el nombre de perceptrón, particularmente cuando se utiliza una función de activación escalón en lugar de una función continua. A lo largo del tiempo, esta arquitectura ha evolucionado significativamente, dando lugar a redes neuronales con múltiples capas y funciones de activación avanzadas, como ReLU, sigmoide, tanh, entre otras, que permiten abordar tareas de mayor complejidad en el aprendizaje automático.

Funciones de activación: Las funciones de activación se aplican inmediatamente después del cálculo de la combinación lineal z , expresada en la ecuación (Ecuación 2.18). Su incorporación en las redes neuronales cumple dos propósitos fundamentales: introducir no linealidad en el modelo y habilitar el uso de técnicas de optimización basadas en gradientes. Dado que el proceso de entrenamiento de una red neuronal consiste en minimizar una función de pérdida e con respecto a los pesos sinápticos w , es necesario que la función de activación $\psi(\cdot)$ sea continua y derivable dentro del dominio de los valores de z . Estas condiciones permiten aplicar algoritmos como el descenso del gradiente y retropropagación (backpropagation) de manera eficiente.

Las funciones de activación determinan cómo se transforma la salida de cada neurona, y su elección depende del tipo de problema que se busca resolver. A continuación, se

describen algunas de las funciones de activación más utilizadas en redes neuronales modernas:

■ **Función ReLU (Rectified Linear Unit):**

La ReLU es de gran utilidad principalmente cuando la tarea a resolver contiene valores numéricos reales positivos, como en el caso del tratamiento de imágenes, esto se debe a que su derivada vale 0 para $z < 0$. Aunque por definición su derivada no existe en cero, se puede determinar su valor a cero o uno en $x = 0$, a su vez, para valores donde $z > 0$ la derivada vale 1.

$$ReLU(z) = \max(0, z) \quad (\text{Ecuación 2.20})$$

- **Función sigmoide logística:** La logística sigmoide es una función en forma de S con un rango de $(0, 1)$, lo cual la hace adecuada para problemas de clasificación binaria, donde se desea una interpretación probabilística de la salida. Su expresión matemática es:

$$\sigma(z) = \frac{1}{1 + \exp(-z)} \quad (\text{Ecuación 2.21})$$

- **Función tangente hiperbólica (tanh):** La función tangente hiperbólica es muy útil ya que su rango es de $(-1, 1)$. Se considera una versión centrada en cero de la sigmoide, lo cual facilita la convergencia durante el entrenamiento. Se puede obtener a partir de la sigmoide como:

$$\tanh(z) = 2\sigma(2z) - 1 = \frac{\exp(z) - \exp(-z)}{\exp(z) + \exp(-z)} \quad (\text{Ecuación 2.22})$$

La tangente hiperbólica es útil en modelos donde se requiere una salida simétrica respecto al origen, y puede acelerar el proceso de aprendizaje en comparación con la sigmoide estándar.

En la Figura 2.10 se ilustra el comportamiento gráfico de las funciones de activación ReLU, sigmoide y tangente hiperbólica, lo que permite observar sus diferencias en términos de salida y pendiente.

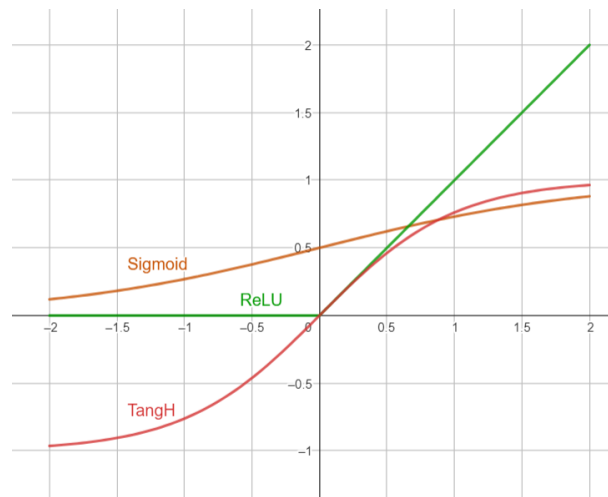


Figura 2.10: Comparación entre las funciones de activación ReLU, sigmoide y tangente hiperbólica. Fuente: Elaboración propia.

La correcta selección de la función de activación tiene un impacto significativo en la velocidad de convergencia del modelo, la profundidad efectiva de la red y su capacidad para generalizar. Por tanto, su elección debe basarse en criterios empíricos, heurísticas derivadas de la literatura, y pruebas experimentales en el contexto particular del problema a resolver.

Función de pérdida: Las funciones de pérdida, también conocidas como funciones de costo o funciones de error, juegan un papel central en el proceso de entrenamiento de modelos de aprendizaje supervisado. Su propósito es cuantificar el grado de discrepancia entre el valor predicho por el modelo, denotado como $\hat{y}$ (véase ecuación (Ecuación 2.19)), y el valor objetivo real y , que representa la etiqueta o salida esperada.

Matemáticamente, la función de pérdida e se define como una función escalar que depende de ambos valores, y y $\hat{y}$, y cuya magnitud indica qué tan bien está funcionando el modelo. Cuanto menor sea el valor de e , más precisa es la predicción.

Sea Q el número total de ejemplos en el conjunto de entrenamiento. Para cada instancia q del conjunto, se calcula una pérdida individual, y el valor final de la función de pérdida se obtiene generalmente como un promedio sobre todas las observaciones.

A continuación se presentan las funciones de pérdida más comunes en tareas de regresión:

- **Error cuadrático medio (MSE):** Esta función penaliza fuertemente los errores grandes al elevar al cuadrado las diferencias entre valores reales y predichos.

$$MSE = \frac{1}{Q} \sum_{q=1}^Q (y^q - \hat{y}^q)^2 \quad (\text{Ecuación 2.23})$$

- **Raíz del error cuadrático medio (RMSE):** Es la raíz cuadrada del MSE. Tiene la ventaja de mantener las mismas unidades que la variable objetivo, lo que facilita la interpretación de la magnitud del error.

$$RMSE = \sqrt{\frac{1}{Q} \sum_{q=1}^Q (y^q - \hat{y}^q)^2} \quad (\text{Ecuación 2.24})$$

- **Error absoluto medio (MAE):** En lugar de elevar al cuadrado las diferencias, esta función toma el valor absoluto, lo que la hace menos sensible a los valores atípicos (*outliers*) en comparación con el MSE. Es útil cuando se desea una medida de error más robusta.

$$MAE = \frac{1}{Q} \sum_{q=1}^Q |y^q - \hat{y}^q| \quad (\text{Ecuación 2.25})$$

La elección de la función de pérdida depende del tipo de problema, la naturaleza de los datos y los objetivos del modelo. En tareas de clasificación, por ejemplo, se suelen utilizar funciones como la entropía cruzada (*cross-entropy*), mientras que en regresión son más comunes las funciones descritas anteriormente.

Un aspecto importante es que las funciones de pérdida deben ser continuas y derivables (al menos casi en todo su dominio), para permitir la aplicación eficiente de algoritmos de optimización basados en el cálculo del gradiente.

Redes neuronales artificiales (ANNs)

Las redes neuronales son aquellos sistemas de procesamiento computacional cuyo objetivo es aprender el conocimiento de diferentes entradas con el fin de optimizar su salida final (Keiron O'Shea, 2015).

Estos modelos se basan en el principio de aprendizaje supervisado y utilizan algoritmos de optimización para ajustar los pesos sinápticos de las conexiones entre neuronas artificiales. A medida que las redes son entrenadas con ejemplos, van mejorando su capacidad de generalización, lo que les permite resolver tareas complejas como clasificación, regresión, reconocimiento de patrones, traducción automática, entre otros. Existen diversas arquitecturas de redes neuronales, cada una adaptada a distintos tipos de problemas. La elección de la arquitectura depende de factores como la naturaleza de los datos (estructurados, imágenes, texto, etc.), el objetivo del modelo, la complejidad del dominio y la cantidad de información disponible para el entrenamiento.

Perceptrón multicapa (MLP) es un tipo de arquitectura donde existen varios perceptrones conectados entre sí, también se le conoce como completamente conectado (*Fully Connected*).

Un MLP está compuesto, en términos generales, por tres tipos de capas:

- Una capa de entrada, que recibe el vector de características x del conjunto de datos X .
- Una o varias capas ocultas, que permiten a la red aprender representaciones intermedias y no lineales.
- Una capa de salida, que entrega la predicción final $\hat{y}$ del modelo.

Cuando la arquitectura contiene múltiples capas ocultas, se considera que forma parte del paradigma de aprendizaje profundo (*deep learning*), el cual ha demostrado un rendimiento notable en tareas complejas gracias a su capacidad para extraer características jerárquicas directamente desde los datos.

La Figura 2.11 muestra un esquema representativo de un MLP de tres capas, en el que N representa el número de neuronas en la capa de entrada, M el número de neuronas en la capa oculta, J la cantidad de neuronas en la capa de salida, que usualmente coincide con el número de clases cuando se trata de un problema de clasificación y Q que indica la cantidad de ejemplos utilizados durante el entrenamiento del modelo.

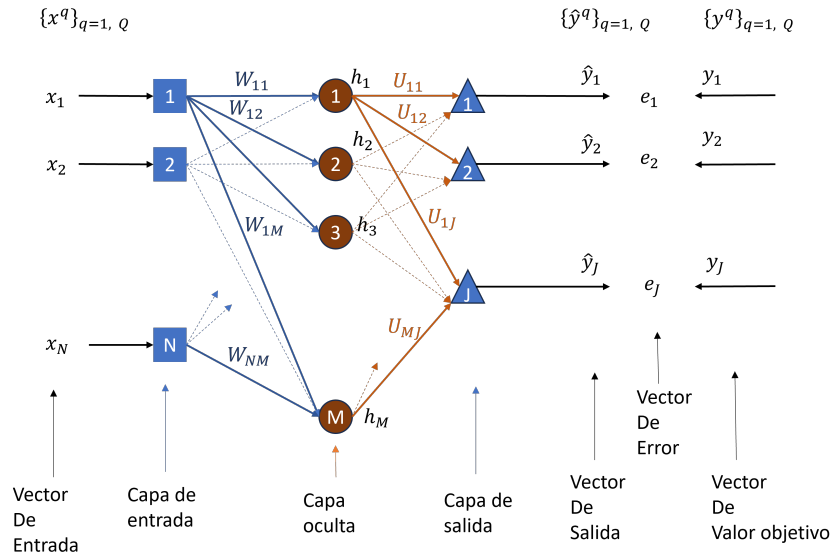


Figura 2.11: Diagrama de Perceptrón multicapa MLP basado en (Looney, 1997).

El entrenamiento es mediante el cual el modelo recibe como entrada un conjunto de datos etiquetados $\{X^q, \mathbf{y}^q\}_{q=1, Q}$. A partir de estos datos, se le muestra a la red la salida esperada, denotada como $\mathbf{y}^q$, correspondiente a cada entrada del conjunto de entrenamiento. El modelo genera una salida estimada $\hat{\mathbf{y}}^q$, y la diferencia entre ambos valores se interpreta como un error $\mathbf{e}^q$, que se desea minimizar.

La minimización de este error se realiza mediante un proceso iterativo de optimización denominado propagación hacia atrás (*backpropagation*), el cual constituye el núcleo del algoritmo de entrenamiento de las redes neuronales. El objetivo de este proceso es ajustar los pesos sinápticos de la red, representados por matrices como W_{NM} (pesos entre la capa de entrada y la capa oculta) y U_{MJ} (pesos entre la capa oculta y la capa de salida), con el fin de mejorar progresivamente la precisión del modelo.

El algoritmo se basa en el cálculo del gradiente de la función de pérdida con respecto a cada uno de los pesos del modelo. Utilizando derivadas parciales, se determinan los

pesos a modificarse para reducir el valor de la función de pérdida. El vector gradiente apunta hacia la dirección de máximo incremento, por lo que el procedimiento sigue su opuesto el descenso del gradiente para moverse hacia el mínimo local o global de la función de error con un avance ponderado conocido como tasa de aprendizaje (*learning rate*) buscando converger a su mínimo.

Con base en la Figura 2.11 las ecuaciones de las salidas de la capa oculta y de salida se determinan como:

$$h_m = \psi_h\left(\sum_{n=1}^N W_{nm}x_n\right) \quad (\text{Ecuación 2.26})$$

$$\hat{y}_j = \psi_{\hat{y}}\left(\sum_{m=1}^M U_{mj}h_m\right) \quad (\text{Ecuación 2.27})$$

Es así como si sustituimos las ecuaciones Ecuación 2.26 y Ecuación 2.27 en la Ecuación 2.23, el error MSE queda de la siguiente forma en términos de los pesos W y U :

$$e = MSE = \frac{1}{Q} \sum_{q=1}^Q \left(\sum_{j=1}^J (y_j^q - \psi_{\hat{y}}\left(\sum_{m=1}^M U_{mj}\psi_h\left(\sum_{n=1}^N W_{nm}x_n^q\right)\right))^2 \right) \quad (\text{Ecuación 2.28})$$

Lo que se busca es encontrar el valor mínimo de la función de costo, para ello se utiliza la siguiente condición de cálculo:

$$\frac{\partial e(W, U)}{\partial W} = 0, \frac{\partial e(W, U)}{\partial U} = 0 \quad (\text{Ecuación 2.29})$$

Sin embargo, la Ecuación 2.29 no se puede resolver en la mayoría de los problemas, su resultado es una aproximación a un valor cercano al mínimo global o local, mediante iteraciones de cálculos que modifiquen los pesos hacia la dirección del gradiente descendente, es por lo que el *backpropagation* debe tener limitaciones que detengan el proceso, como lo es una tolerancia de aceptación en el error o un número máximo de iteraciones, también conocidas como épocas (Looney, 1997).

El gradiente está determinado por:

$$\nabla e(W, U) = \left(\frac{\partial e(W, U)}{\partial W}, \frac{\partial e(W, U)}{\partial U} \right) \quad (\text{Ecuación 2.30})$$

La aproximación de los nuevos pesos queda determinada por:

$$(W, U)^{r+1} = (W, U)^r - \eta(\nabla e(W, U)) \quad (\text{Ecuación 2.31})$$

donde $\eta > 0$ es la tasa de aprendizaje o *learning rate*. Un valor alto ocasiona que con los nuevos pesos calculados el resultado se pase del valor mínimo, y si es un valor muy pequeño, se necesitarían muchas iteraciones para llegar al mínimo. El parámetro r representa el índice de iteración dentro del proceso de entrenamiento de la red neuronal. En la iteración inicial ($r = 0$), los pesos sinápticos suelen ser inicializados aleatoriamente, típicamente utilizando distribuciones uniformes o normales con media cero, o bien, mediante esquemas más elaborados como la inicialización de Xavier o He, dependiendo de la arquitectura del modelo y la función de activación utilizada. Esta aleatoriedad en la inicialización es crucial para romper la simetría entre las neuronas y permitir un aprendizaje efectivo desde las primeras etapas del entrenamiento.

Redes neuronales convolucionales (CNN) son una arquitectura especializada que ha demostrado gran eficacia en el procesamiento de datos con estructuras espaciales o temporales, tales como imágenes bidimensionales (2D) o señales unidimensionales (1D) como texto o audio. Estas redes se caracterizan por su capacidad para extraer automáticamente patrones locales mediante la aplicación de filtros o kernels, reduciendo así la necesidad de ingeniería manual de características. Este modelo esta representado por:

El modelo matemático simplificado de la operación convolucional en dos dimensiones se representa como:

$$Y[i, j] = \sum_{m=0}^{K-1} \sum_{n=0}^{K-1}, x[i \cdot s + m - p, j \cdot s + n - p] \cdot W[m, n] \quad (\text{Ecuación 2.32})$$

donde:

$Y[i, j]$ es el valor en la posición (i, j) del mapa de salida resultante,

$x[a, b]$ Valor de la entrada (imagen o mapa de características) en la posición (a, b) ,

$W[m, n]$ es el kernel o filtro convolucional en la posición (m, n) ,

K tamaño del filtro $K \times K$,

m, n índices de recorrido sobre el filtro,

s representan el desplazamiento o *stride* aplicado,

p es el *padding* o cantidad de ceros añadidos alrededor de la entrada,

i, j índices espaciales que recorren la salida Y .

Aunque la operación descrita en (Ecuación 2.32) corresponde a una correlación cruzada, en la práctica se le conoce comúnmente como convolución, dado que el filtro no es rotado (a diferencia de la definición matemática clásica de convolución). El stride determina cuántas posiciones se desplaza el filtro entre cada operación, afectando directamente la resolución espacial del mapa de salida.

La Figura 2.12 ilustra visualmente cómo opera una convolución sobre una imagen, mostrando la interacción del filtro con distintas regiones de entrada.

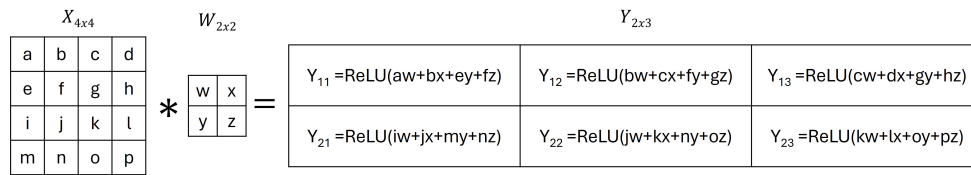


Figura 2.12: Ejemplo ilustrativo del proceso de convolución con un filtro deslizante. El número de celdas recorridas entre cada operación está determinado por el parámetro *stride*.

Cuando la matriz de salida Y generada por las operaciones de convolución es de gran tamaño, su procesamiento puede volverse computacionalmente costoso y menos eficiente. Para abordar este problema, se implementa una operación conocida como *pooling*. Esta técnica explota la invarianza local, es decir, la capacidad del modelo para identificar características importantes independientemente de su ubicación exacta dentro de una región vecina.

La operación de *pooling* consiste en recorrer la matriz Y con una ventana de tamaño fijo por ejemplo, de 2×2 y seleccionar el valor máximo, promedio o mínimo dentro de cada región. De este modo, se genera una nueva matriz con menor resolución espacial, pero que retiene las características más significativas del área analizada. Este proceso no solo reduce la complejidad computacional del modelo, sino que también contribuye a mitigar el sobreajuste (*overfitting*), al eliminar información redundante. La Figura 2.13 ilustra el funcionamiento de esta operación sobre un mapa de activación considerando el máximo.

$$Output = \begin{array}{|c|c|} \hline \max(Y_{11}, Y_{12}, Y_{21}, Y_{22}) & \max(Y_{12}, Y_{13}, Y_{22}, Y_{23}) \\ \hline \end{array}$$

Figura 2.13: Ejemplo del proceso de *max pooling* con ventana de 2×2 . En cada región se selecciona el valor máximo.

Una red convolucional típica contiene múltiples filtros o kernels, cada uno diseñado para extraer un tipo específico de patrón (bordes, texturas, formas, etc.). La salida generada por cada kernel se conoce como canal o *feature map*, y al finalizar las capas convolucionales, todas estas salidas son concatenadas y aplanadas en un único vector unidimensional. Este vector es posteriormente enviado a una red completamente conectada (*Multilayer Perceptron, MLP*), que actúa como clasificador final.

El MLP puede estar compuesto por una o varias capas ocultas, dependiendo de la complejidad del problema, y su objetivo es interpretar las representaciones aprendidas por la CNN y asignarlas a una clase específica. Este diseño modular permite combinar el poder de extracción automática de características de las redes convolucionales con la capacidad predictiva de los modelos densamente conectados ([Raschka y cols., 2022](#)).

Redes neuronales recurrentes (RNN) son otro tipo de red neuronal en el cual la salida generada funciona como entrada a la misma red. Esta propiedad les permite mantener una memoria interna del estado anterior, lo que resulta especialmente útil en tareas donde el orden y el contexto de los datos son fundamentales.

Este tipo de red es ideal para problemas de regresión y clasificación sobre secuencias,

como el procesamiento de texto, predicción de series temporales, análisis de audio o traducción automática. De forma intuitiva, se puede imaginar una RNN como una rotación de una MLP en 90° en sentido contrario a las manecillas del reloj, lo que permite extender su funcionalidad a lo largo de múltiples pasos de tiempo.

La Figura 2.14 ilustra el esquema de una RNN desdoblada en el tiempo. En este modelo, a cada paso temporal t , se procesa una entrada $x^{(t)}$ y se genera un estado oculto $h^{(t)}$, el cual depende tanto de la entrada actual como del estado oculto anterior $h^{(t-1)}$. El estado oculto es entonces utilizado para calcular una salida $o^{(t)}$ en ese mismo instante.

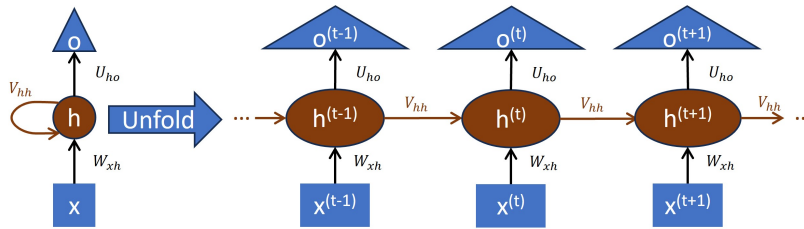


Figura 2.14: Modelo esquemático de una RNN con desdoblamiento temporal (*unfold*).

Las ecuaciones que describen al modelo de la RNN en la figura 2.14 son las siguientes:

$$h^{(t)} = \psi_h(W_{xh}x^{(t)} + W_{hh}h^{(t-1)} + b_h) \quad (\text{Ecuación 2.33})$$

$$o^{(t)} = \psi_o(W_{ho}h^{(t)} + b_o) \quad (\text{Ecuación 2.34})$$

donde:

$x^{(t)}$ es la entrada en el tiempo t ,

$h^{(t)}$ es el estado oculto en el tiempo t ,

$o^{(t)}$ es la salida producida,

W_{xh} , W_{hh} , W_{ho} son matrices de pesos sinápticos,

b_h y b_o son los términos de sesgo (bias),

$\psi_h(\cdot)$ y $\psi_o(\cdot)$ son funciones de activación (típicamente tanh o ReLU para ψ_h y softmax o lineal para ψ_o).

Este tipo de red neuronal implica el uso del tiempo ya que la predicción se hace con base en los valores ya obtenidos.

Analizando la Figura 2.14, se observa que el proceso de entrenamiento en una red neuronal recurrente (RNN) no puede ser paralelizado de forma eficiente, ya que la salida de cada instante de tiempo t depende del estado oculto del paso anterior $h^{(t-1)}$. Esta dependencia secuencial impide que las operaciones se realicen de forma simultánea y obliga a que el entrenamiento se realice de manera iterativa y en serie.

Para entrenar este tipo de redes, se utiliza un algoritmo denominado retropropagación a través del tiempo (*Backpropagation Through Time, BPTT*), el cual extiende el algoritmo tradicional de retropropagación para considerar los distintos pasos temporales de la secuencia. En *BPTT*, se desdobra la red a lo largo del tiempo (como en la Figura 2.14), se acumulan los errores de cada instante, y se realiza la retropropagación considerando la dependencia de los estados pasados, lo que permite actualizar los pesos globales del modelo.

Una variable crítica en el diseño de RNNs es la longitud del contexto, es decir, cuántas salidas o estados anteriores se consideran relevantes para realizar una predicción. Un contexto muy largo implica una red muy profunda en el tiempo, lo que conlleva desafíos técnicos significativos, en particular los problemas conocidos como el desvanecimiento y la explosión del gradiente (*vanishing and exploding gradients*), ampliamente discutidos en la literatura ([Goodfellow, Bengio, Courville, y Bach, 2017](#)).

Estos problemas surgen por el comportamiento exponencial de la multiplicación repetida de la matriz de pesos recurrentes W_{hh} durante el entrenamiento:

Si $W_{hh} < 1$, las derivadas parciales tienden a disminuir exponencialmente con cada paso de tiempo, lo que provoca que el gradiente se vuelva insignificante y dificulte el aprendizaje de relaciones a largo plazo (desvanecimiento del gradiente).

Por el contrario, si $W_{hh} > 1$, el gradiente puede crecer exponencialmente y volverse inestable, generando actualizaciones extremadamente grandes (explosión del gra-

diente).

Idealmente, se buscaría que la magnitud de W_{hh} se mantuviera alrededor de 1 para preservar el flujo de gradientes, pero en la práctica esto es difícil de garantizar durante todo el entrenamiento. Para abordar estos problemas, se han propuesto arquitecturas alternativas más robustas, como las Redes de Memoria a Largo y Corto Plazo (Long Short-Term Memory, LSTM), las cuales incorporan celdas de memoria internas que sustituyen al estado oculto tradicional de las RNNs y permiten preservar información útil durante periodos más prolongados de tiempo ([Raschka y cols., 2022](#)).

Celdas de memoria a largo y corto plazo (LSTM): fue introducida por Sepp Hochreiter y Jürgen Schmidhuber en 1997, con el objetivo de superar las limitaciones de las RNN tradicionales, particularmente el problema del desvanecimiento del gradiente. Las LSTM han demostrado una alta eficacia en el modelado de dependencias a corto y largo plazo dentro de secuencias, lo que las convierte en una de las arquitecturas más utilizadas en tareas de procesamiento de lenguaje natural, series temporales, y secuencias de audio o video.

A diferencia de las RNN estándar, las LSTM incorporan una estructura interna más compleja que deja atrás el concepto tradicional de neurona artificial. En su lugar, introducen una unidad de memoria, comúnmente referida como celda, la cual está compuesta por múltiples compuertas que regulan el flujo de información. Estas compuertas permiten al modelo decidir de forma dinámica qué información conservar, qué olvidar y qué actualizar, lo que les confiere una notable capacidad para manejar secuencias largas sin degradación significativa en el aprendizaje.

La Figura 2.15 muestra el esquema funcional de una celda LSTM, donde se observa la interacción de las compuertas internas y los flujos de datos entre ellas.

Las ecuaciones que describen el comportamiento de una LSTM en el paso de tiempo t se muestran en la ecuación (Ecuación 2.35):

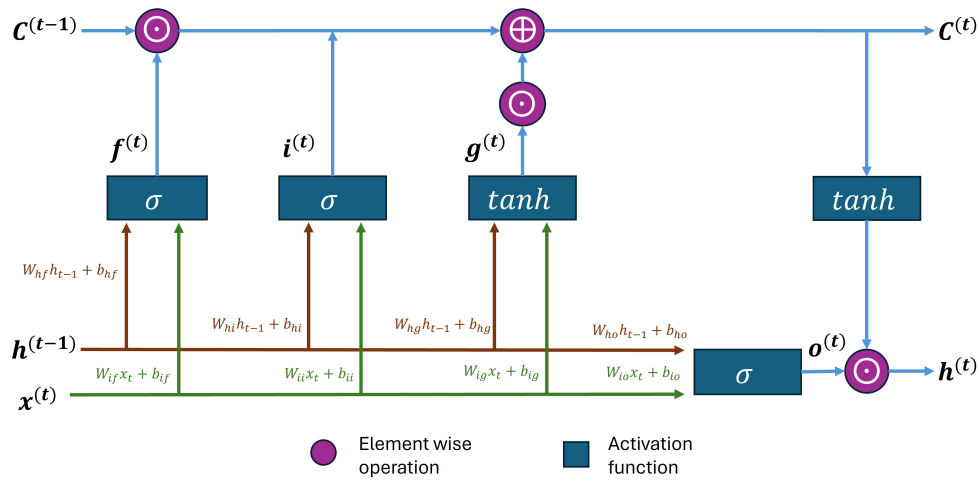


Figura 2.15: Modelo esquemático de una celda LSTM. Las compuertas controlan la actualización y preservación de la memoria a lo largo del tiempo.

$$\begin{aligned}
 f^{(t)} &= \sigma(W_{if}x^{(t)} + b_{if} + W_{hf}h^{(t-1)} + b_{hf}) && \text{(compuerta de olvido)} \\
 i^{(t)} &= \sigma(W_{ii}x^{(t)} + b_{ii} + W_{hi}h^{(t-1)} + b_{hi}) && \text{(compuerta de entrada)} \\
 g^{(t)} &= \tanh(W_{ig}x^{(t)} + b_{ig} + W_{hg}h^{(t-1)} + b_{hg}) && \text{(información candidata)} \\
 C^{(t)} &= (C^{(t-1)} \odot f^{(t)}) \oplus (i^{(t)} \odot g^{(t)}) && \text{(estado de la celda)} \\
 o^{(t)} &= \sigma(W_{io}x^{(t)} + b_{io} + W_{ho}h^{(t-1)} + b_{ho}) && \text{(compuerta de salida)} \\
 h^{(t)} &= o^{(t)} \odot \tanh(C^{(t)}) && \text{(estado oculto)}
 \end{aligned}
 \tag{Ecuación 2.35}$$

Donde:

$x^{(t)}$ es la entrada en el instante t .

$h^{(t-1)}$ es el estado oculto anterior.

$f^{(t)}$ es la compuerta de olvido, que determina cuánta información del estado previo $C^{(t-1)}$ se conserva.

$i^{(t)}$ es la compuerta de entrada, que regula cuánta información nueva se agrega a la memoria.

$g^{(t)}$ representa la información candidata a ser añadida a la celda.

$C^{(t)}$ es el estado interno de la celda, que actúa como memoria a largo plazo.

$o^{(t)}$ es la compuerta de salida, que decide qué parte del estado de la celda se transmite como salida.

$h^{(t)}$ es la salida final o estado oculto de la unidad en el tiempo t .

σ representa la función sigmoide, y $\tanh$ la tangente hiperbólica.

$\odot$ y $\oplus$ indican multiplicación y suma elemento a elemento, respectivamente (operaciones Hadamard).

Esta arquitectura permite que las LSTM aprendan a preservar información útil durante muchos pasos temporales, lo que las hace significativamente más efectivas que las RNN estándar en tareas de contexto largo, sin requerir una arquitectura excesivamente profunda ni mecanismos adicionales de corrección de gradientes.

Unidad Recurrente Cerrada (GRU): Esta arquitectura fue propuesta en el año 2014 por Kyunghyun Cho y colaboradores, como una variante simplificada de las LSTM. El modelo GRU conserva la capacidad de manejar dependencias de corto y largo plazo en secuencias, pero con una estructura interna más eficiente y computacionalmente menos costosa.

A diferencia de las LSTM, las GRU eliminan la necesidad de una celda de memoria y reducen el número de compuertas, fusionando las funciones de la compuerta de entrada y de olvido en una sola denominada compuerta de actualización. Además, no requieren una compuerta de salida ni una activación adicional para producir la salida final, lo cual simplifica tanto la implementación como el entrenamiento. Este diseño permite que las GRU mantengan una eficiencia similar a la de las LSTM, pero con menos parámetros y menor consumo computacional.

La Figura 2.16 muestra un esquema de una celda GRU según la implementación adoptada por el *framework* PyTorch ([Paszke y cols., 2019](#)). Aunque han surgido diversas variantes de GRU a lo largo de los años, la versión presentada corresponde a la más común y ampliamente utilizada en la práctica.

Existen diferentes tipos de GRU que varían en la arquitectura de la celda, y que han cambiado con los años. El modelo representado en la Figura 2.16 es el modelo que

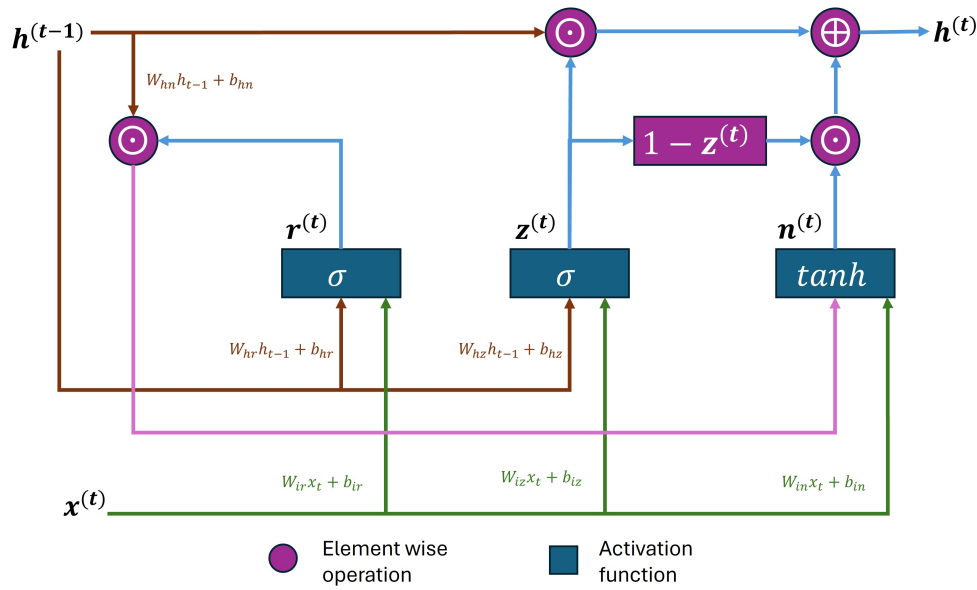


Figura 2.16: Modelo esquemático de la GRU en PyTorch. Fuente: (Paszke y cols., 2019)

se encuentra disponible en el *framework* *PyTorch* (Paszke y cols., 2019).

Las ecuaciones que definen el comportamiento de la GRU son las siguientes:

$$\begin{aligned}
 z^{(t)} &= \sigma(W_{iz}x^{(t)} + b_{iz} + W_{hz}h^{(t-1)} + b_{hz}) && \text{(compuerta de actualización)} \\
 r^{(t)} &= \sigma(W_{ir}x^{(t)} + b_{ir} + W_{hr}h^{(t-1)} + b_{hr}) && \text{(compuerta de reinicio)} \\
 n^{(t)} &= \tanh(W_{in}x^{(t)} + b_{in} + r^{(t)} \odot (W_{hn}h^{(t-1)} + b_{hn})) && \text{(estado propuesto)} \\
 h^{(t)} &= (1 - z^{(t)}) \odot n^{(t)} + z^{(t)} \odot h^{(t-1)} && \text{(estado oculto final)}
 \end{aligned}$$

(Ecuación 2.36)

Donde:

$x^{(t)}$ es la entrada en el tiempo t ,

$h^{(t-1)}$ es el estado oculto del paso anterior,

σ representa la función sigmoide,

$\tanh$ es la función tangente hiperbólica,

$\odot$ denota la multiplicación elemento por elemento (producto de Hadamard),

$z^{(t)}$ es la compuerta de actualización, que decide cuánta información del estado anterior debe mantenerse,

$r^{(t)}$ es la compuerta de reinicio, que controla qué parte del estado anterior se utiliza para generar el nuevo contenido,

$n^{(t)}$ es el nuevo contenido candidato para el estado actual,

$h^{(t)}$ es el estado oculto actualizado de la celda.

Gracias a su simplicidad estructural, las GRU han ganado popularidad en aplicaciones donde el rendimiento computacional es una preocupación, como en dispositivos móviles o tareas en tiempo real. Además, como lo señalan Ekman et al. ([Ekman, 2021](#)) y otros investigadores, los resultados empíricos indican que no existe una diferencia significativa en el desempeño entre LSTM y GRU en muchas tareas prácticas, por lo que la elección entre ambas arquitecturas suele depender de criterios computacionales o experimentales.

Las redes neuronales recurrentes (RNN) y sus variantes más avanzadas, como LSTM y GRU, representan herramientas fundamentales en el procesamiento de datos secuenciales. Si bien las RNN tradicionales constituyen la base de esta clase de modelos, su limitada capacidad para retener información en secuencias largas debido al desvanecimiento del gradiente las hace inadecuadas para tareas complejas.

Las LSTM surgieron como una solución eficaz a esta limitación, al incorporar una celda de memoria con compuertas que permiten aprender dependencias de largo plazo. Por su parte, las GRU simplifican esta estructura manteniendo un rendimiento competitivo, lo que las convierte en una alternativa práctica cuando se busca un equilibrio entre precisión y eficiencia computacional.

La elección entre estas arquitecturas debe basarse en el tipo de problema, la longitud y naturaleza de las secuencias, la disponibilidad de recursos computacionales y los requisitos de tiempo de entrenamiento. En resumen, las LSTM ofrecen mayor capacidad de modelado, mientras que las GRU destacan por su simplicidad y eficiencia. Ambas superan claramente a las RNN estándar en la mayoría de aplicaciones reales de secuencias temporales y texto.

Para resumir las principales diferencias entre las arquitecturas recurrentes descritas, en la Tabla 2.1 se presenta una comparación detallada entre las redes RNN, LSTM y

GRU, considerando aspectos como la estructura, capacidad de aprendizaje a largo plazo, complejidad computacional y casos de uso recomendados.

Tabla 2.1: Comparación entre RNN, LSTM y GRU

Característica	RNN	LSTM	GRU
Fecha de origen	1980s	1997	2014
Control del flujo de información	No posee compuertas internas	Compuertas de entrada, olvido y salida	Compuerta de actualización y reinicio
Manejo de dependencia a largo plazo	Deficiente (problemas de gradiente)	Muy bueno	Bueno, comparable a LSTM
Complejidad computacional	Baja	Alta	Media
Cantidad de parámetros	Pocos	Muchos	Menos que LSTM
Velocidad de entrenamiento	Rápida, pero ineficaz	Lenta (más cálculos por celda)	Más rápida que LSTM
Recomendado para	Secuencias cortas o simples	Secuencias largas y complejas	Soluciones rápidas y eficientes con buena precisión

Ajuste de hiperparámetros

Como se ha descrito en secciones anteriores, el rendimiento de una red neuronal depende no solo de la arquitectura seleccionada, sino también de un conjunto de valores predefinidos que controlan el proceso de aprendizaje. Estos valores, conocidos como hiperparámetros, deben establecerse antes de iniciar el entrenamiento y tienen un impacto significativo en la capacidad del modelo para generalizar adecuadamente sobre datos no vistos.

Entre los hiperparámetros más relevantes se encuentran: el tipo de red neuronal (MLP, CNN, RNN, etc.), el número de capas ocultas, la cantidad de neuronas por capa, las funciones de activación utilizadas, la tasa de aprendizaje (learning rate), los métodos de inicialización de pesos, la función de pérdida, el número de épocas de entrenamiento, el tamaño del batch, entre otros. La elección adecuada de estos parámetros puede ser determinante para obtener un modelo eficiente y robusto.

Dado que no existe una regla general para definir los valores óptimos de los hiperparámetros, se recurre al uso de técnicas de búsqueda que permiten explorarlos sistemáticamente. A continuación, se describen las dos estrategias más utilizadas para este propósito:

Búsqueda por malla (*Grid Search*): La técnica de búsqueda por malla consiste en definir un conjunto discreto de valores posibles para cada hiperparámetro y evaluar todas las combinaciones posibles entre ellos. Aunque garantiza una exploración exhaustiva del espacio definido, este enfoque se vuelve computacionalmente costoso cuando el número de hiperparámetros o el rango de valores es elevado. Por ello, su uso suele limitarse a escenarios donde se manejan pocos parámetros o cuando se cuenta con una alta capacidad de cómputo (Ekman, 2021).

Búsqueda aleatoria (*Random Search*): A diferencia del enfoque anterior, la búsqueda aleatoria evalúa un número determinado de combinaciones generadas aleatoriamente dentro del espacio definido de hiperparámetros. Esta técnica es más eficiente cuando el número de parámetros es alto, ya que puede explorar regiones del espacio más relevantes sin necesidad de evaluar todas las combinaciones posibles. Es ampliamente utilizada en la práctica debido a su balance entre eficiencia computacional y calidad de resultados (Géron, 2019).

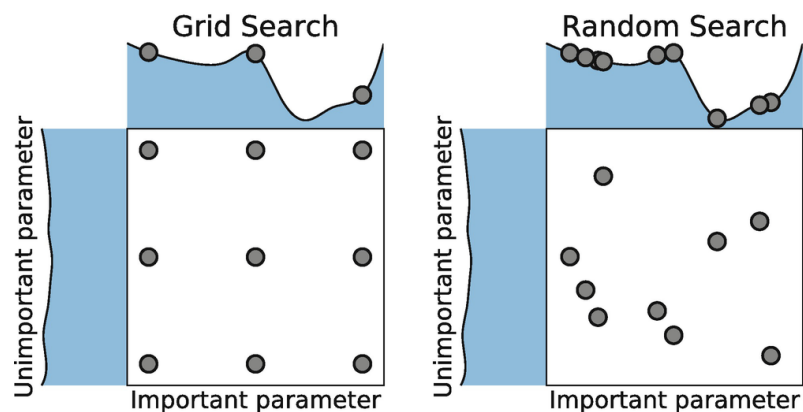


Figura 2.17: Comparación gráfica entre Grid Search (izquierda) y Random Search (derecha). Fuente: <https://velog.velcdn.com/images/crosstar1228/post/86ed2303-4861-4f30-8738-1ada436b4e32/image.png>

Durante el proceso de búsqueda, cada combinación de hiperparámetros es evaluada mediante validación cruzada o sobre un conjunto de validación específico, registrando tanto los valores evaluados como los pesos del modelo entrenado. Esto permite seleccionar la configuración que presenta el mejor desempeño según las métricas definidas, y guardar los pesos asociados al mejor modelo encontrado para su uso posterior.

El ajuste de hiperparámetros es, por tanto, una etapa crítica en el desarrollo de modelos de aprendizaje profundo, ya que tiene un impacto directo en la capacidad del modelo para aprender de manera efectiva y generalizar adecuadamente en datos reales.

2.2. Estado del arte

Para el desarrollo de esta investigación, resulta fundamental considerar la temporalidad de las ventas registradas en el negocio. En este sentido, ([Fernando, 2021](#)) llevó a cabo un estudio enfocado en la predicción de ventas de motocicletas mediante el uso de series temporales. El autor empleó el modelo ARTXP, el cual presentó el menor margen de error en comparación con otros modelos evaluados. El desarrollo de dicho estudio se realizó utilizando la herramienta *Microsoft Analysis Services*. No obstante, en el contexto de esta investigación, se toman como referencia los modelos aplicados en la valuación de autopartes (ARIMA, MIXED, ARTXP), a fin de analizar sus respectivos desempeños en series temporales.

Uno de los desafíos recurrentes en la predicción de ventas surge con los productos de reciente lanzamiento, ya que no se dispone de datos históricos suficientes para realizar estimaciones confiables. En este contexto, ([Smirnov y Sudakov, 2021](#)) identificó que los árboles de decisión pueden representar una solución viable ante esta limitación. En su estudio, se evaluaron tres modelos: XGBoost, LightGBM y CatBoost, siendo LightGBM el que obtuvo el mejor desempeño. La propuesta central consistió en utilizar variables categóricas de productos similares ya existentes en el mercado, con el propósito de aproximar el comportamiento esperado de ventas para los nuevos artículos.

La investigación realizada por (Bhat, 2023) destaca que SAP, como sistema de gestión empresarial, constituye una herramienta valiosa para la administración eficiente del *scrap*. No obstante, el autor señala ciertas limitaciones inherentes al sistema, como la necesidad de ingresar datos de forma manual, lo que puede afectar la precisión y eficiencia del proceso. Asimismo, se plantea la oportunidad de futuras investigaciones orientadas a cerrar la brecha existente en la integración de SAP con otros elementos clave de la manufactura, tales como el inventario de productos y la planeación de la producción. Este último aspecto será abordado en el presente trabajo.

Dos meses después de la propuesta de (Bhat, 2023) sobre la necesidad de investigar la relación entre la planeación de la producción y el *scrap*, (Bode, Goetz, Schleich, y Wartzack, 2023) plantearon una posible asociación entre el *scrap* y las fallas en las máquinas. Su hipótesis principal establece que un aumento en el desperdicio puede preceder a una falla operativa. Para analizar esta relación, se empleó una metodología multiobjetivo, la cual permite identificar correlaciones considerando múltiples características simultáneamente. Esta metodología resulta pertinente para el presente estudio, ya que puede aplicarse al análisis de los datos disponibles con el objetivo de detectar patrones similares.

Asimismo, es fundamental considerar las distintas técnicas disponibles para la planeación de la producción, especialmente cuando se busca una gestión en tiempo casi real. En este sentido, (Matonya, Kocsi, Pusztai, y Budai, 2020) realizaron una investigación orientada a analizar los factores previos que deben ser considerados antes de implementar un sistema de Automatización Robótica de Procesos (*Robotic Process Automation, RPA*). Dicho estudio resulta relevante para el presente trabajo, ya que las técnicas propuestas pueden integrarse en el diseño del sistema automático de planeación de la producción que se plantea como producto final de esta investigación.

A diferencia de lo propuesto por (Matonya y cols., 2020), (Krishnan, Khan, y Al-Qurni, 2022) incorporaron técnicas de planeación de la producción más recientes, que integran tecnologías de inteligencia artificial (*IA*) y aprendizaje automático. En su estudio, los autores destacan que las metodologías tradicionales se basaban en esquemas de planeación estática; no obstante, la adopción de *IA* permite una planeación dinámica,

capaz de adaptarse a cambios imprevistos en el entorno productivo. Además, se introducen los conceptos de Planeación Agregada de Producción (*Aggregate Production Planning, APP*) y Sistemas Ciberfísicos (*Cyber-Physical Systems, CPS*), los cuales están siendo implementados por empresas manufactureras de vanguardia.

([Lu, Shang, Xu, Shao, y Zhang, 2023](#)) propusieron el uso del modelo BiGRU para la predicción de ventas de autopartes, específicamente tornillos. GRU es una arquitectura de red neuronal recurrente mejorada que aborda eficazmente los problemas de desaparición y explosión del gradiente durante el proceso de entrenamiento. En el modelo GRU convencional, la predicción se basa únicamente en la información proveniente del pasado (*backward layer*), mientras que el modelo BiGRU incorpora además una capa que considera información futura (*forward layer*), lo que mejora significativamente la capacidad de aprendizaje contextual. Como resultado, el modelo BiGRU alcanzó una precisión superior, con un coeficiente de determinación R^2 de 0.99. Adicionalmente, al comparar las predicciones con datos reales correspondientes a un periodo de 28 días, se obtuvo un R^2 de 0.96, lo que confirma su desempeño robusto.

La combinación de arquitecturas representa una línea de investigación activa en el campo del aprendizaje profundo, tanto para la extracción de características como para tareas de inferencia. La implementación de modelos híbridos ha demostrado ser eficaz en diversas aplicaciones, como la predicción de ventas de productos perecederos. En este contexto, ([Chen y cols., 2024](#)) propusieron un modelo combinado GRU-LightGBM, el cual permite optimizar la comercialización de estos productos a través de plataformas en línea, minimizando el tiempo de almacenamiento en bodegas y reduciendo el riesgo de entregas de productos caducados al cliente.

Diversas investigaciones han demostrado que la arquitectura LSTM ofrece un desempeño superior en tareas de predicción de ventas en comparación con otras redes neuronales; no obstante, este modelo requiere una mayor cantidad de ejemplos para un aprendizaje efectivo, así como un mayor costo computacional. Asimismo, el preprocesamiento de los datos del conjunto de entrenamiento resulta crucial. Por ejemplo, la eliminación de valores atípicos que se desvíen de una norma preestablecida, junto

con técnicas de normalización como *Z-score*, contribuyen a acelerar el proceso de entrenamiento. En el estudio realizado por ([Mogarala Guruvaya, Kollu, Divakarachari, Falkowski-Gilski, y Praveena, 2024](#)), se utilizó un modelo GRU bidireccional acompañado de una estrategia avanzada de ingeniería de características, la cual incluyó el método de redundancia mínima y relevancia máxima (*Minimum Redundancy Maximum Relevance, MRMR*), la eliminación recursiva de características (*Recursive Feature Elimination, RFE*) y la optimización adaptativa mediante enjambre de partículas (*Adaptive Particle Swarm Optimization, APSO*). Estas técnicas contribuyeron a reducir la necesidad de altos costos de almacenamiento y mejoraron el rendimiento del modelo. El resultado obtenido fue un coeficiente de correlación de 0.98 sobre un conjunto de datos correspondiente a Walmart.

Los modelos GRU y LSTM suelen ser comparables en cuanto a rendimiento; sin embargo, presentan arquitecturas distintas, lo que justifica su evaluación individual frente a problemáticas específicas. Existen investigaciones orientadas a comparar el desempeño de distintos algoritmos recurrentes en la predicción de ventas de productos. De acuerdo con los resultados presentados por ([Riatma y cols., 2025](#)), el modelo LSTM supera ligeramente al GRU en términos de precisión. No obstante, cuando se requieren modelos menos complejos computacionalmente y con menor cantidad de parámetros por aprender, la arquitectura GRU se posiciona como una alternativa eficiente, ofreciendo resultados aceptables con menor costo computacional.

Capítulo 3

Metodología

En la Figura 3.1, ubicada en la página 54, se presenta el diagrama esquemático de la metodología empleada en el desarrollo de esta tesis. Dicho diagrama resume de manera clara y estructurada las distintas etapas del proceso de investigación, proporcionando una guía visual para la ejecución ordenada de cada fase del proyecto. La metodología adoptada sigue una secuencia lógica que abarca desde la adquisición y preparación de los datos, hasta la construcción, validación y evaluación de los modelos predictivos, lo que garantiza un flujo de trabajo coherente, sistemático y reproducible.

3.1. Obtención de datos

Los datos utilizados en esta investigación fueron proporcionados por una empresa privada del sector automotriz. La información corresponde al historial de ventas facturadas desde el año 2018 hasta 2024, obtenida mediante consultas al sistema de planificación de recursos empresariales SAP ([SAP, 2024](#)). Adicionalmente, para el mismo periodo, se recopilaron registros relacionados con el proceso de producción, tales como el número de piezas fabricadas y la cantidad de desperdicio generado durante la manufactura.

Las ventas fueron obtenidas mediante el uso de la transacción KE24, perteneciente al módulo de Control de Costos (*Controlling*, CO) del sistema SAP. Esta transacción

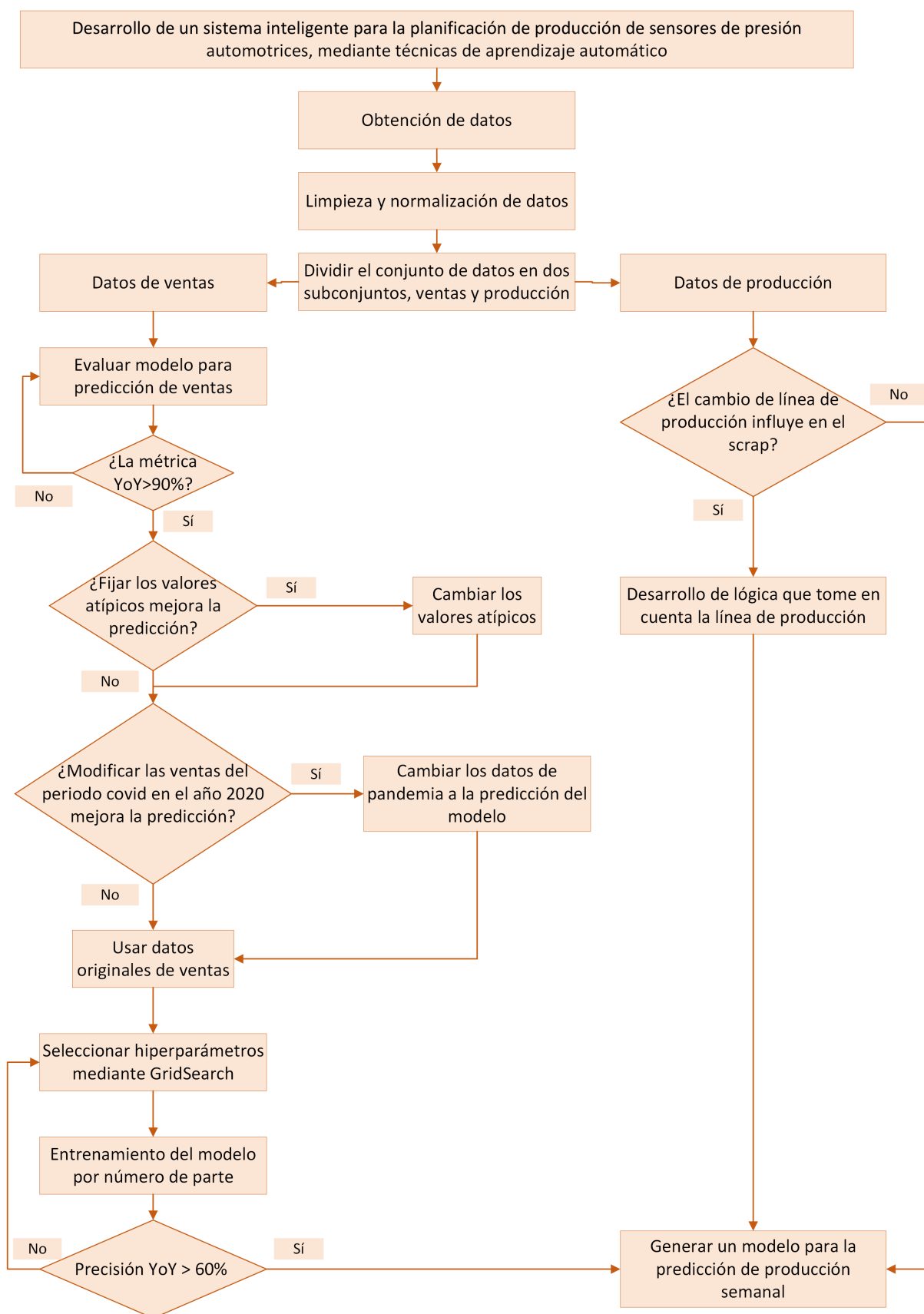


Figura 3.1: Diagrama de flujo del método

permite visualizar los elementos individuales de ventas en formato de lista detallada. Para recuperar únicamente la información relevante, se configuraron los filtros necesarios dentro de los campos del formulario, restringiendo la consulta a los números de parte de interés para esta investigación.

Para optimizar el proceso de consulta, se utilizaron símbolos conocidos como *wild cards*, los cuales permiten agilizar el llenado de los parámetros de búsqueda. Esto se debe a que los números de parte se registran con una longitud de 13 dígitos, aunque, para efectos de esta investigación, únicamente los primeros 10 dígitos son relevantes. Por ejemplo, si se desea recuperar la información correspondiente a los números de parte A123456789-123 y A123456789-456, es suficiente con solicitar los registros asociados a A123456789*, donde el asterisco (*) actúa como comodín (*wild card*) para cubrir cualquier combinación de los últimos dígitos.

La información relacionada con la producción fue extraída mediante la transacción MB51 del sistema SAP. Para la consulta, los números de parte se filtraron utilizando únicamente los primeros diez dígitos, dado que los registros de producción se generan con esta longitud. Adicionalmente, se aplicó un filtro por tipo de movimiento, correspondiente a las confirmaciones de producción: el código 131 indica confirmaciones positivas, mientras que el 132 representa cancelaciones. Esta clasificación permite calcular el total de materiales efectivamente producidos..

El material desperdiciado fue obtenido mediante la transacción FCOREP_SCRAP, la cual permite cuantificar tanto en unidades como en términos monetarios el impacto del material que no cumple con los estándares de fabricación establecidos y, por lo tanto, debe ser desechado. Al igual que en los casos anteriores, se seleccionaron los números de parte utilizando los primeros diez dígitos correspondientes a los artículos que se desean analizar.

En las tres transacciones descritas, se estableció como periodo de posteo el intervalo comprendido entre el 1 de enero de 2018 y el 31 de diciembre de 9999 (formato: 01.01.2018 al 31.12.9999). Este rango permite recuperar la información registrada desde el inicio del histórico disponible hasta la fecha de extracción de los datos.

Con el objetivo de resguardar la confidencialidad de la empresa, los valores sensibles

han sido omitidos u ocultados en la Figura 3.2, la cual muestra un ejemplo del archivo descargado. Cabe señalar que los encabezados se presentan en idioma inglés, al igual que el código fuente compartido en el siguiente apartado, debido a que así lo exige la política interna de la empresa propietaria de los datos.

La predicción de ventas se llevará a cabo utilizando tres columnas clave: *PostingDate*, *Product* y *Quantity*, las cuales corresponden, respectivamente, a la fecha de facturación, el número de parte del sensor y la cantidad vendida. Por motivos de privacidad y seguridad, los identificadores de producto han sido anonimizados en todos los análisis presentados en esta tesis.

PostingDate	Plant	Product	Country	Customer	Bill_to_Party	Ship_to_Party	Quantity	TotalSale
04/01/2018	Oculto	Oculto	Oculto	Oculto	Oculto	Oculto	1200	6881.46027

Figura 3.2: Datos de ventas obtenida de la empresa de autopartes. Se han eliminado los valores sensibles

3.2. Limpieza y normalización de datos

Antes de realizar cualquier análisis o modelado, los datos deben ser sometidos a un proceso riguroso de limpieza y preparación. Este procedimiento contempla la identificación y corrección de errores tales como valores atípicos, datos nulos, inconsistencias y registros con información no válida o ilógica. Con el fin de facilitar su tratamiento, los datos fueron organizados en dos conjuntos principales: uno correspondiente a las ventas y otro relativo al proceso de producción.

Se utilizaron comentarios dentro del código con el propósito de describir detalladamente la función de cada comando implementado (Flores, 2024a). A continuación, se presenta el script en lenguaje Python empleado para procesar el archivo original extraído mediante la transacción KE24, el cual genera como salida el conjunto de datos consolidado en el archivo *SalesbyPN.csv*.

```

1  #Importación de librerías
2  import pandas as pd
3  import matplotlib.pyplot as plt
4  from datetime import datetime as dt

```

```

5  #Crear dataframe del archivo original de KE24
6  data=pd.read_csv('Sales.csv') #Lectura de archivo
7  data = data.astype(str) #Leer todas las columnas como texto
8  data.Product = 'PN' + data.Product.astype(str) #Agregar PN al inicio
9  data['isProduct']=data['Product'].apply(lambda x:len(x)) #Agrega la longitud
    del SKU
10 data=data[data['isProduct']==15] #Solamente los materiales de venta son
    analizados
11 data.drop(data.columns[[1,3,4,5,6,-1]],axis=1,inplace=True)#Elimina columnas
    sin usar
12 #Formateo de campos
13 data.PostingDate=pd.to_datetime(data['PostingDate'], format='%Y-%m-%d') #Tipo
    fecha
14 data.Quantity=pd.to_numeric(data['Quantity'],downcast='integer') #Tipo entero
15 data.TotalSale=pd.to_numeric(data['TotalSale'],downcast='float') #Tipo
    flotante
16
17 data=data[data['Quantity']>0].reset_index().rename(columns={'Product': "PN"})#
    Solo los valores positivos son analizados
18 data=data.groupby(['PostingDate','PN']).agg({'Quantity':'sum','TotalSale':'sum
    '}).reset_index()#Se agrupan los datos para tener solo un dato de venta por
    fecha
19 dates = pd.date_range('2018-01-01', '2023-12-31') # Rango de fechas a
    completar
20 pn_index=pd.DataFrame(pn2023).rename(columns={0: "PN"})
21 pn_index.set_index('PN',inplace=True,drop=False)
22 # Acompletar todas las fechas para cada SKU
23 pn_index=pn_index.apply(
24     lambda x: dates, axis=1
25 ).explode().reset_index(name='PostingDate')[['PostingDate', 'PN']]
26 #Crear dataframe completo
27 df=pn_index.merge(data,how='left', on=['PostingDate','PN'])#Union
28 df['Quantity'] = df['Quantity'].fillna(0) #Rellenar con cero
29 df['TotalSale'] = df['TotalSale'].fillna(0) #Rellenar con cero
30
31 #adding date fields
32 df['PN10']=df.PN.str[:12] #Trim the SKU
33 df['Year']=df.PostingDate.dt.year

```



```
34 df['Month']=df.PostingDate.dt.month
35 df['Week']=df.PostingDate.dt.isocalendar().week
36 df['DayofWeek']=df.PostingDate.dt.dayofweek
37 df.to_csv('SalesbyPN.csv')
```

Descripción del procesamiento aplicado:

El código presentado tiene como objetivo estandarizar, limpiar y estructurar el conjunto de datos de ventas extraído desde SAP, a fin de prepararlo para su análisis y modelado. Las principales transformaciones realizadas son las siguientes:

- **Estandarización del identificador de producto:** Se antepone el prefijo 'PN' a cada número de parte con el fin de unificar su formato, lo cual facilita su identificación y análisis posterior.
- **Filtrado por longitud del SKU:** Se seleccionan únicamente aquellos productos cuyo identificador tiene una longitud de 15 caracteres, dado que esta condición corresponde a los materiales comercializados. Esta restricción permite excluir registros que no pertenecen al catálogo de ventas válidas.
- **Eliminación de columnas irrelevantes:** Se suprimen columnas no utilizadas para reducir la dimensionalidad del conjunto de datos y optimizar el uso de memoria.
- **Conversión de tipos de datos:** Se transforma el tipo de dato de las columnas clave (fecha, cantidad y monto de venta) a sus respectivos formatos numéricos o temporales, lo cual es necesario para operaciones posteriores como agregaciones o modelado.
- **Filtrado de ventas válidas:** Se descartan registros con cantidades negativas o igual a cero, asegurando que el análisis se centre exclusivamente en ventas efectivas.
- **Agregación por fecha y producto:** Se agrupan los datos por número de parte y fecha de facturación para obtener una única entrada por día y producto, consolidando las cantidades y montos vendidos. Una de las tareas más relevantes en

esta etapa fue la completitud temporal de los datos. En SAP, los días sin actividad como los domingos sin ventas no se registran de forma explícita, lo que genera huecos en la secuencia temporal. Para garantizar la continuidad de las series, se insertaron registros con valor cero en todas las fechas faltantes para cada número de parte, asegurando así un formato uniforme y compatible con los algoritmos de predicción.

- **Relleno de valores faltantes:** Una vez generada la estructura temporal completa, se imputaron ceros en los campos de cantidad y monto de venta en los días sin transacciones registradas, evitando errores durante el entrenamiento de modelos.
- **Generación de nuevas variables temporales:** Se extrajeron características adicionales de la fecha, tales como el año, mes, semana y día de la semana, que serán útiles para la agrupación en visualizaciones y análisis posteriores.

Finalmente, el conjunto de datos procesado se exportó en formato `.csv` bajo el nombre `SalesbyPN.csv`, el cual se utilizará en las siguientes etapas del análisis.

Este archivo constituye la base de datos final utilizada para la etapa de predicción de ventas. En la Figura 3.3 se presenta un extracto del archivo procesado, junto con el fragmento de código correspondiente y su respectiva descripción técnica.

B	C	D	E	F	G	H	I	J
PostingDate	PN	Quantity	TotalSale	PN10	Year	Month	Week	DayofWeek
01/01/2018	Oculto	0	0	Oculto	2018	1	1	0

Figura 3.3: Archivo de ventas preparado para su uso en el modelo de predicción

Con el objetivo de optimizar los tiempos de entrenamiento y evaluar el desempeño del modelo de manera más eficiente, se decidió restringir el análisis a los diez números de parte con mayor volumen de ventas durante el año 2023.

3.3. Evaluar modelo para predicción de ventas

Para la implementación del modelo de predicción se empleó una arquitectura basada en redes GRU, utilizando la librería desarrollada por (Olivares, Challú, Garza, Canseco, y

[Dubrawski, 2022](#)). Considerando las limitaciones de tiempo y el alcance definido para esta tesis, se optó por enfocar el análisis exclusivamente en dicha arquitectura, sin realizar comparaciones extensivas con otros modelos. Esta decisión se fundamenta en estudios previos, como el de ([Ekman, 2021](#)), quien documenta que el desempeño de las redes GRU es comparable al de las LSTM en diversas tareas de predicción secuencial, con la ventaja de requerir menor complejidad computacional ya que se evalúa obtener un modelo entrenado GRU para cada número de parte.

Antes de aplicar el modelo para la predicción individual de cada número de parte, se realizó un análisis preliminar empleando la métrica de variación interanual (*Year-over-Year*, YoY), definida en la ecuación (Ecuación 2.11), así como la evaluación del tiempo de ejecución requerido para las etapas de entrenamiento y predicción. Como criterio mínimo de viabilidad, se estableció que el modelo debe alcanzar una precisión YoY superior al 90%.

3.4. Cambio de datos originales de las ventas nulas durante el covid

Durante el periodo comprendido entre marzo y mayo del año 2020, las ventas registradas fueron igual a cero, debido al cierre temporal de operaciones causado por la pandemia de COVID-19. Este comportamiento se considera un evento atípico, que no refleja el patrón habitual de comportamiento del sistema de producción ni de la demanda. Dado que los modelos de aprendizaje automático están diseñados para generalizar el comportamiento regular de los datos, es pertinente evaluar el impacto de reemplazar dichos valores atípicos por estimaciones generadas por el propio modelo.

El objetivo de este análisis es verificar si la sustitución de los datos nulos correspondientes al periodo de pandemia mejora la capacidad predictiva del modelo para años posteriores, específicamente para el año 2023.

Previo a dicha modificación, se realizará una evaluación comparativa del desempeño

3.4. CAMBIO DE DATOS ORIGINALES DE LAS VENTAS NULAS DURANTE EL COVID

del modelo al mantener o ajustar los valores considerados atípicos, reconociendo que picos o caídas abruptas suelen representar un desafío para modelos entrenados bajo supuestos de continuidad o estacionalidad.

Con este propósito, se definen los siguientes cuatro escenarios experimentales:

- **Escenario 1:** Predicción del periodo de COVID utilizando los datos originales sin modificaciones.
- **Escenario 2:** Predicción del periodo de COVID eliminando los valores atípicos detectados.
- **Escenario 3:** Predicción del año 2023 utilizando todo el historial de ventas original, incluyendo el periodo de COVID sin ajustes.
- **Escenario 4:** Predicción del año 2023 reemplazando los valores del periodo de COVID con estimaciones generadas por el modelo.

El análisis de estos escenarios permitirá determinar si la intervención sobre los datos históricos mejora la capacidad del modelo para capturar patrones representativos y realizar predicciones más precisas en contextos no anómalos.

3.4.1. Código base para el tratamiento de datos

El código utilizado para esta etapa del procesamiento se encuentra disponible en ([Flores, 2024b](#)) para su consulta o reutilización.

Dado que en el archivo `SalesPN.csv` se completaron las fechas sin ventas, el siguiente procedimiento consiste en dividir el conjunto de datos en dos subconjuntos: uno para entrenamiento y otro para validación.

La función a utilizar es la siguiente presentada:

```
1 def set_train_valid(df,ini,fin,div):  
2     '''  
3     input: df: dataframe a separar  
4     ini: fecha de inicio
```

3.4. CAMBIO DE DATOS ORIGINALES DE LAS VENTAS NULAS DURANTE EL COVID

```
5     fin: fecha de termino
6     div: fecha de división, considerada en la sección de validación
7     output: train: dataframe para entrenamiento
8     valid: dataframe para validación
9
10    '''
11    df=df[['PostingDate','Quantity','PN10','Year','Month','Week','DayofWeek']]
12    df=df.rename(columns={'PostingDate': 'ds', 'Quantity': 'y', 'PN10': 'unique_id
13                        '})#Nixtla requiere estos nombres
14    df=df.loc[df['ds'] >= ini]
15    df=df.loc[df['ds'] <= fin]
16    df=df.groupby(['ds','unique_id','Year','Month','Week','DayofWeek'])['y'].sum()
17    .reset_index()
18    train = df.loc[df['ds'] < div]
19    valid = df.loc[df['ds'] >= div]
20    return train,valid
```

La función `set_train_valid` tiene como propósito dividir el conjunto de datos en dos subconjuntos: entrenamiento y validación, con base en una fecha de corte definida por el usuario. Esta función recibe como argumentos un *dataframe* con los datos procesados, una fecha de inicio (`ini`), una fecha de término (`fin`) y una fecha de división (`div`) que determina la frontera temporal entre los subconjuntos.

El procedimiento incluye los siguientes pasos:

- Se seleccionan únicamente las columnas relevantes: fecha de publicación (`PostingDate`), cantidad vendida (`Quantity`), identificador del producto a 10 dígitos (`PN10`) y variables derivadas de la fecha.
- Las columnas son renombradas para cumplir con el formato requerido por la librería `Nixtla`, donde:
 - `ds` representa la dimensión temporal
 - `y` la variable objetivo,
 - `unique_id` el identificador de serie temporal por producto.

- Se filtran los datos para incluir únicamente el rango de fechas comprendido entre `ini` y `fin`.
- Se realiza una agregación por fecha, identificador único y atributos temporales, sumando los valores de venta por día y producto.
- Finalmente, los datos se dividen en dos subconjuntos:
 - `train`: incluye los registros anteriores a la fecha `div`.
 - `valid`: incluye los registros desde `div` en adelante.

Esta función permite estructurar los datos de forma compatible con modelos de series temporales y facilita la evaluación del rendimiento predictivo mediante un conjunto de validación temporalmente coherente.

3.5. Selección de hiperparámetros

Con el objetivo de mejorar la precisión del modelo de predicción, se optó por entrenar un modelo independiente para cada uno de los números de parte considerados en el estudio. Esta decisión se fundamenta en que el comportamiento temporal de las ventas es específico para cada producto, y modelar cada serie individualmente permite capturar de forma más precisa sus patrones únicos, lo que incrementa la efectividad global del sistema de pronóstico.

Para el ajuste de hiperparámetros, se empleó la técnica de búsqueda en malla (Grid Search, véase página 47), la cual permite evaluar de manera sistemática y controlada múltiples combinaciones de parámetros relevantes para cada modelo. Este enfoque garantiza la trazabilidad de los experimentos y facilita la comparación objetiva entre configuraciones bajo criterios homogéneos.

```
1 h = valid['ds'].nunique() #Fechas unicas
2 hiperparam={
3     'h': [h],
4     'max_steps': [5120, 1024, 512], #Epocas
```

```
5     'encoder_activation': ['relu'], #Función de activación
6     'encoder_n_layers':[4,2], #Capas de entrada
7     'encoder_hidden_size':[512], #Neuronas capa oculta
8     'context_size':[1], #Historico de contexto
9     'encoder_dropout':[0.5], #Probabilidad de olvido
10    'decoder_hidden_size':[128], #Tamaño de capa de salida
11    'decoder_layers':[2], #Capas de salida
12    'learning_rate':[0.001], #Tasa de aprendizaje
13    'scaler_type':['robust'], #Escalador de datos
14    'loss':[
15        DistributionLoss(distribution='Poisson', level=[90]),
16        MAE(), MSE(),
17        RMSE()
18    ],
19    }#Función de pérdida
```

Con base en los resultados obtenidos en términos de precisión y tiempo de ejecución, se podrá definir una metodología general escalable. Esto resulta especialmente útil en caso de que, en el futuro, se requiera extender el modelo de predicción al total del catálogo de productos, permitiendo replicar el procedimiento de forma eficiente y consistente a nivel industrial.

3.6. Entrenamiento del modelo por número de parte

Una vez seleccionada la configuración óptima de hiperparámetros, se procede a entrenar el modelo utilizando la totalidad de los datos disponibles hasta el año 2023, con el fin de generar predicciones para ese periodo. El objetivo es que el modelo alcance una precisión interanual (*Year-over-Year*, YoY), definida en la ecuación (Ecuación 2.11), superior al 60%, y que el error absoluto medio (*Mean Absolute Error*, MAE), descrito en la ecuación (Ecuación 2.25), sea minimizado. Estos dos criterios permiten evaluar tanto la capacidad del modelo para seguir la tendencia general del negocio como su exactitud en las estimaciones puntuales.

El entrenamiento se realiza de forma independiente para cada número de parte y

para cada combinación de hiperparámetros definida en el proceso de búsqueda. Si el resultado de una predicción supera al de las iteraciones anteriores en términos de las métricas establecidas, el modelo correspondiente es almacenado para su uso posterior en la etapa de predicción final.

3.6.1. Código base para el entrenamiento

El proceso de entrenamiento y evaluación de modelos ha sido estructurado en módulos reutilizables con el fin de facilitar su mantenimiento, interpretación y escalabilidad. Cada uno de los archivos desarrollados cumple una función específica en el flujo de trabajo.

Definición del modelo

El modelo GRU utilizado se encuentra definido en el archivo `Models.py`. Este archivo contiene una función que recibe como argumentos los hiperparámetros seleccionados y devuelve una instancia del modelo con las configuraciones requeridas.

```
1 from neuralforecast.models import GRU
2 def neural_model (**kwargs):
3     models = [GRU(futr_exog_list = ['Month', 'Week', 'DayofWeek'],
4         **kwargs)]
5     return models
```

Entrenamiento y predicción del modelo

El archivo `Testing.py` contiene la función encargada de entrenar el modelo y generar predicciones sobre el conjunto de validación. Esta función utiliza la librería `NeuralForecast` y devuelve tanto el modelo entrenado como un *dataframe* con las predicciones, el cual se fusiona con los datos reales para facilitar la evaluación posterior.

```
1 import importlib
2 from neuralforecast import NeuralForecast #Libreria de uso
3 import Models # Se importan los modelos a utilizar (GRU)
```



```

4 importlib.reload(Models)
5 def test(train, valid, hiper, type='gru'):
6     model = NeuralForecast(models = Models.neural_model(**hiper), freq='D')
7     model.fit(train) # Entrenamiento del modelo
8     p = model.predict(futr_df=valid) #Predicción del modelo ya entrenado
9     p = p.merge(valid[['ds', 'unique_id', 'y']], on=['ds', 'unique_id'], how='left
10     ')#Unión con dataframe original
11     return model, p

```

Evaluación del modelo

Para evaluar la calidad de las predicciones generadas, se emplean distintas métricas de desempeño, definidas en el archivo `Pred_eval.py`. Estas métricas incluyen el error absoluto medio (MAE), el coeficiente de determinación (R^2), la razón de ventas predichas vs reales (YoY), y el coeficiente de correlación de Pearson.

```

1 from sklearn.metrics import mean_absolute_error, r2_score
2 def valuate_prediction_loss(p):
3     loss = mean_absolute_error(p['y'], p['GRU'])
4     r2 = r2_score(p['y'], p['GRU'])
5     year=p['GRU'].sum()/p['y'].sum()
6     pearson=p['y'].corr(p['GRU'])
7     return {'Year':year, 'MAE':loss, 'R2':r2, 'Pearson':pearson}

```

Almacenamiento del modelo entrenado

El archivo `CreateDir.py` se encarga de crear las carpetas necesarias para almacenar los modelos entrenados. Si el desempeño de una nueva combinación de hiperparámetros supera al modelo anterior, este se guarda en su correspondiente directorio.

```

1 from pathlib import Path
2 def save_model(folder='models', pn='default'):
3
4     # 1. Crea carpeta general
5     MODEL_PATH = Path(folder)
6     MODEL_PATH.mkdir(parents=True, exist_ok=True)

```

```
7
8 # 2. Crea carpeta específica
9 MODEL_NAME = pn
10 MODEL_SAVE_PATH = MODEL_PATH / MODEL_NAME
11
12 MODEL_PATH = Path(MODEL_SAVE_PATH)
13 MODEL_PATH.mkdir(parents=True, exist_ok=True)
```

Automatización del proceso de entrenamiento y búsqueda de hiperparámetros

La automatización del proceso completo se lleva a cabo en el archivo `AutoML.py`. Este módulo permite realizar la búsqueda en malla (Grid Search) para cada número de parte, entrenar y evaluar todos los modelos candidatos, y guardar aquel con mejor desempeño de acuerdo con la métrica seleccionada (por defecto, MAE).

El algoritmo garantiza que los modelos evaluados sean trazables y que se registre el desempeño de cada configuración. Los resultados se almacenan en archivos CSV por número de parte.

```
1 import pandas as pd
2 import numpy as np
3 import os
4
5 import Testing
6 import Pred_Eval
7 import importlib
8 import CreateDir
9
10 importlib.reload(Testing)
11 importlib.reload(Pred_Eval)
12 importlib.reload(CreateDir)
13
14 def product(*args):
15     '''Makes the combination for GridSearch'''
16     pools = [pool for pool in args[0]]
17     result = [[]]
18     for pool in pools:
```

```

19     result = [x+[y] for x in result for y in pool]
20     for prod in result:
21         yield list(prod)
22
23 def tuning_params(train,
24                   valid,
25                   hyperparam,
26                   type = 'gru',
27                   folder='models',
28                   metric='MAE'):
29     '''
30     Search best hyperparameters by grid search
31     input: train: dataframe trainset
32     valid: dataframe validset
33     hyperparam: dict for gridsearch
34     folder: folder to save models
35     '''
36     assert train['unique_id'].nunique() == valid['unique_id'].nunique(), '
37         unique_id numbers must match between train and valid sets'
38     for pn in train['unique_id'].unique():
39         t = train[train['unique_id']==pn]
40         v = valid[valid['unique_id']==pn]
41         mae = float('inf')
42         path=folder + '/' + pn
43         for r in product(list(hyperparam.values())):
44             saved=[0]
45             hiperparam=dict(zip(hyperparam.keys(),r))
46             print(f'PN: {pn} and Evaluating: {hiperparam}')
47             model_trained,p=Testing.test(train=t,
48                                         valid=v,
49                                         hiper=hiperparam,
50                                         type='gru'
51                                         )
52             d = Pred_Eval.valuate_prediction_loss(p)
53             if d['MAE']<mae:
54                 mae = d['MAE']
55                 saved = [1]
56             print('*****Saving model*****')

```

```

56     print(f'*PN: {pn} \n*Hyperparams: {hiperparam} \n*Stats:{d}\n')
57     CreateDir.save_model(folder=folder,pn=pn)
58     model_trained.save(path=path,
59                        model_index=None,
60                        overwrite=True,
61                        save_dataset=True)
62     cond = os.path.isfile(path+'/doe.csv')
63     if cond:
64         df = pd.read_csv(path+'/doe.csv', header = None)
65         #print(df.iloc[:,1])
66         df = df[ df.iloc[:,1]!='h']
67         df = df[ df.iloc[:,1]!=np.nan]
68         df.iloc[:,-1:] = 0
69         #print(df)
70         df.to_csv(path+'/doe.csv', header = False, index = False)
71         df = pd.DataFrame([pn] + r + list(d.values()) + saved).transpose()
72         #print(df)
73         df.to_csv(path+'/doe.csv', header = False, index = False, mode='a')
74         cols=['PN'] + list(hyperparam.keys()) + list(d.keys()) + ['Best Model']
75         #print('Columnas: ', cols)
76         df = pd.read_csv(path+'/doe.csv', header = None)
77         df.columns = cols
78         df.to_csv(path+'/doe.csv', index = False)
79         print('\n\n*****')

```

Evaluación por escenarios experimentales

Una vez implementado y validado el sistema de entrenamiento, los modelos generados pueden ser evaluados en los distintos escenarios definidos en la Sección 3.4. Esto permite comparar el impacto de distintas decisiones sobre el preprocesamiento de los datos como el tratamiento de outliers durante la pandemia de COVID-19 y establecer cuál configuración ofrece el mejor equilibrio entre precisión y robustez.

3.6.2. Código para la evaluación de escenarios

Una vez tratados los datos conforme a lo descrito en la Sección 3.5 y almacenados en formato .csv, es posible realizar la evaluación de los escenarios definidos en la Sección 3.4. Este análisis se lleva a cabo en un cuaderno de Jupyter, específicamente en el archivo 2023_covid_filled_shared.ipynb.

```

1 import torch
2 import importlib
3 import pandas as pd
4 import TrainSplit
5 import AutoML
6 from neuralforecast.losses.pytorch import DistributionLoss, MSE, RMSE, MAE
7
8 # Cargar y procesar archivo principal
9 data = pd.read_csv('Analysis/SalesbyPN.csv')
10 data.PostingDate=pd.to_datetime(data['PostingDate'], format='%Y-%m-%d')
11 data.Quantity=pd.to_numeric(data['Quantity'],downcast='integer')
12 df=data
13
14 # Filtrar los materiales relevantes
15 tenLeadTypes=['']#list of materials
16 lead_types= tenLeadTypes
17 df=df[df['PN10'].isin(lead_types)]

```

Escenario 1

El primer escenario evalúa el comportamiento del modelo durante el periodo de pandemia sin realizar modificaciones a los datos originales. A continuación, se muestra el fragmento de código correspondiente a la configuración de fechas, división de los datos y ejecución del proceso de búsqueda de hiperparámetros:

```

1 ini = '2018-01-01'      # Fecha de inicio del histórico
2 fin = '2020-02-28'      # Fecha límite antes del inicio del impacto por COVID-19
3 div = '2020-01-01'      # Fecha de división entre entrenamiento y validación
4
5 # Generación de subconjuntos de entrenamiento y validación

```

```

6 train, valid = TrainSplit.set_train_valid(df, ini, fin, div)
7
8 # Carpeta para almacenar los modelos evaluados en este escenario
9 folder = 'escenario_1'
10
11 # Búsqueda de hiperparámetros y entrenamiento
12 AutoML.tunning_params(train=train2,
13                       valid=valid2,
14                       hyperparam=hiperparam,
15                       type = 'gru',
16                       folder=folder)

```

Escenario 2

En este escenario se evalúa el impacto de reducir los valores atípicos detectados en las series temporales de ventas. La estrategia aplicada consiste en reemplazar aquellos valores extremos que superan el límite superior del rango definido por el diagrama de caja (*boxplot*) anual, y no en eliminarlos completamente del conjunto de datos. Es decir, los valores atípicos son sustituidos por el máximo valor permitido según la lógica del bigote superior de cada año, lo que constituye una reducción controlada más que una eliminación directa.

Este procedimiento se implementa en el script `Clean_outliers.py`, el cual contiene las siguientes funciones:

```

1 import matplotlib.pyplot as plt
2
3 # Obtención del valor máximo permitido por año y producto
4 def max_values(p, by='Year'):
5     max_vals = {}
6     for pn in p['unique_id'].unique():
7         plot_df = p[p['unique_id']==pn]
8         fig = plt.figure(figsize=(2,5), layout='constrained')
9         df = plot_df[[by, 'y']]
10        data=[]
11        for _ in df[by].unique():

```

```

12     d=df[df[by]==_]['y']
13     data.append(d)
14     box = plt.boxplot(data)
15     plt.close()
16     max_whisker = []
17     for i in range(len(df[by].unique()*2):
18         if i%2!=0:
19             max_whisker.append(box['caps'][i].get_ydata()[0])
20     max_vals[pn] = max_whisker
21     return max_vals
22
23
24 # Reemplazo de outliers por el máximo permitido según boxplot
25 def cleaning(p, by='Year'):
26     d = max_values(p=p,by=by)
27     print(d)
28     for pn in p['unique_id'].unique():
29         print(pn)
30         for year in range(len( p['Year'].unique() )):
31             maxi=d[pn][year]
32             year2=p['Year'].unique()[year]
33             p.loc[(p['unique_id']==pn) & (p['Year']==year2) & (p['y']>maxi), ['y']] =
                 maxi
34     return p

```

Una vez aplicado el ajuste, la estructura del proceso de entrenamiento y evaluación es la misma que en el Escenario 3.6.2, con la única diferencia de que los datos utilizados han sido preprocesados para contener valores atípicos suavizados.

La única modificación necesaria en el flujo de código es la actualización del nombre de la carpeta donde se almacenarán los modelos evaluados para este escenario:

```

1     folder = 'escenario_2'

```

Escenario 3

Este escenario evalúa el desempeño del modelo al utilizar el historial completo de ventas, incluyendo el periodo afectado por la pandemia de COVID-19, sin aplicar ningún tipo de ajuste o tratamiento a los valores atípicos. El objetivo es establecer una línea base para comparar el impacto de no intervenir los datos históricos, en contraste con los escenarios que sí aplican correcciones.

A continuación se muestra el código correspondiente a la configuración del periodo de entrenamiento y validación, así como a la ejecución del proceso de ajuste de hiperparámetros mediante búsqueda en malla:

```
1  ini = '2018-01-01'      # Inicio del periodo de entrenamiento
2  fin = '2023-12-31'     # Fin del histórico disponible
3  div = '2023-01-01'     # Punto de corte para validación
4
5  # División de los datos
6  train, valid = TrainSplit.set_train_valid(df, ini, fin, div)
7
8  # Carpeta donde se almacenarán los resultados del escenario 3
9  folder = 'escenario_3'
10
11 # Ejecución del entrenamiento y selección de hiperparámetros
12 AutoML.tunning_params(train=train2,
13                        valid=valid2,
14                        hyperparam=hiperparam,
15                        type = 'gru',
16                        folder=folder)
```

En este escenario se espera que la presencia de datos anómalos especialmente los correspondientes al periodo de pandemia afecte el desempeño del modelo, lo cual será contrastado con los resultados obtenidos en el Escenario 3.6.2 (sin periodo COVID) y el Escenario 4 (con reemplazo de valores).

Escenario 4

En este escenario se evalúa el impacto de reemplazar los valores correspondientes al periodo de pandemia por estimaciones generadas previamente por el modelo GRU. Esta estrategia busca mitigar el efecto de datos atípicos o no representativos como aquellos registrados entre marzo y mayo de 2020, al sustituirlos por predicciones más alineadas con el comportamiento general de la serie.

El procedimiento inicia con la carga del archivo `covid_prediction.csv`, el cual contiene los valores pronosticados para el periodo en cuestión. Posteriormente, se eliminan los registros reales correspondientes a los meses de marzo, abril y mayo de 2020, y se reemplazan por los valores estimados:

```

1 # Cargar predicción generada previamente para el periodo COVID
2 covid = pd.read_csv('covid_pred_best/covid_prediction.csv')
3 covid = covid.drop(['y'], axis=1)
4 covid['y'] = covid['GRU'].round().abs()
5 covid = covid.drop(['GRU'], axis=1)
6 covid.ds = pd.to_datetime(covid.ds, format='%Y-%m-%d')
7 covid.y = pd.to_numeric(covid.y, downcast='integer')
8
9 # Remover registros originales del periodo COVID
10 train_no_covid = train.copy()
11 train_no_covid['id'] = train_no_covid['Year'].astype('str') + train_no_covid['
    Month'].astype('str')
12 rows_to_drop = ['20203', '20204', '20205']
13 train_no_covid = train_no_covid.loc[~train_no_covid['id'].isin(rows_to_drop)]
14 train_no_covid = train_no_covid.drop(['id'], axis=1)
15
16 # Sustituir por predicciones
17 train_covid_filled = train_no_covid.append(covid)

```

Una vez realizado el reemplazo, se continúa con el proceso habitual de entrenamiento y evaluación, almacenando los resultados en una carpeta específica para este escenario:

```

1 train2 = train_covid_filled
2 valid2 = valid

```

```
3 folder = 'escenario_4'
4 import AutoML
5 importlib.reload(AutoML)
6 AutoML.tunning_params(train=train2,
7                        valid=valid2,
8                        hyperparam=hiperparam,
9                        type = 'gru',
10                       folder=folder)
```

Este enfoque permite analizar si la intervención directa sobre los datos históricos mejora la capacidad del modelo para capturar patrones más representativos, sin verse afectado por eventos atípicos como lo fue la interrupción operativa durante la pandemia.

3.7. Desarrollo de lógica que tome en cuenta la línea de producción

Uno de los objetivos complementarios de esta tesis es identificar posibles relaciones entre el producto fabricado, la línea de producción en la que se procesa y la generación de desperdicio (*scrap*). Determinar esta interacción resulta relevante tanto para el diseño de sistemas de planeación más precisos como para la toma de decisiones operativas basadas en datos.

Con este propósito, se propone un análisis exploratorio inicial con enfoque visual, empleando la herramienta de visualización de datos *Power BI*. Esta aproximación permitirá detectar patrones o correlaciones entre variables clave, tales como el tipo de sensor, la línea de producción asignada y la proporción de piezas descartadas durante el proceso de manufactura. A partir de estos hallazgos, se busca definir una lógica de planeación que incorpore las restricciones y variaciones reales del entorno productivo.

El primer paso para generar estas visualizaciones consiste en importar los datos al entorno de *Power BI*. Para ello, una vez instalada la herramienta, se debe acceder al panel de inicio, seleccionar la opción Obtener datos y posteriormente elegir el formato

3.8. Generar un modelo que evalúe la predicción de producción

Una vez estimada la demanda de ventas, cuantificado el nivel de desperdicio (*scrap*) generado en el proceso y definida la capacidad máxima de producción por línea, es posible establecer un modelo de aprendizaje profundo que determine la cantidad óptima de piezas a fabricar. Este modelo, basado en la arquitectura GRU (véase página 44), tiene como propósito garantizar que la demanda del cliente sea cubierta de manera oportuna, considerando tanto las pérdidas por desperdicio como las restricciones operativas del entorno productivo.

El objetivo es calcular una producción planificada ajustada, que permita cumplir con los compromisos comerciales sin sobreutilizar recursos ni generar excedentes innecesarios. De esta manera, se busca integrar la predicción de ventas con la lógica del proceso de manufactura de manera eficiente y realista.

Para estimar el impacto del *scrap*, se plantea utilizar como referencia el promedio de desperdicio registrado en los últimos 12 meses o bien aplicar un modelo de regresión lineal para proyectar dicho valor. A continuación, se describe el procedimiento de cálculo, implementado en lenguaje Python.

Preparación de datos de producción

Primero se filtran los registros de producción correspondientes al año más reciente (2023), considerando únicamente las líneas de producción relevantes y los productos de interés:

```
1 import pandas as pd
2 import numpy as np
3 cost_centers = # Lista de líneas de producción
4 tenLeadTypes= #Lista de números de parte
5 pcs_produced = pd.read_csv('pcs_produced.csv')
6 pcs_produced = pcs_produced[pcs_produced['Cost Center'].notnull()]
7 pcs_produced['Cost Center'] = pd.to_numeric(pcs_produced['Cost Center'],
      downcast='integer')
```

```
8 pcs_produced = pcs_produced[pcs_produced['Len']==10]
9 pcs_produced = pcs_produced[pcs_produced['Product Flag']=='EZ']
10 pcs_produced = pcs_produced[pcs_produced['Cost Center'].isin(cost_centers)]
11 pcs_produced = pcs_produced[pcs_produced['Year']==2023]
12 lead_types_produced = pcs_produced[pcs_produced['Material'].isin(tenLeadTypes)].
    copy()
```

Preparación de datos de desperdicio

En paralelo, se cargan y filtran los registros de *scrap*, eliminando entradas no relevantes:

```
1 scrap = pd.read_csv('scrap.csv', encoding='latin-1')
2 scrap = scrap[scrap['Quantity'].notnull()]
3 scrap = scrap[scrap['Type']!='MISC']
4 scrap['Cost Center'] = pd.to_numeric(scrap['Cost Center'], downcast='integer')
5 scrap = scrap[scrap['Cost Center'].isin(cost_centers)]
6 scrap = scrap[scrap['Year']==2023]
7 lead_types_scrapped = scrap[scrap['Material'].isin(tenLeadTypes)].copy()
8 lead_types_scrapped.rename(columns={'Period':'Month'}, inplace=True)
```

Cálculo del porcentaje de desperdicio

Se agrupan los datos de producción y desperdicio por mes, y se calcula el porcentaje mensual de desperdicio para cada número de parte:

```
1 cols = ['Material', 'Year', 'Month']
2 lead_types_produced_group = pd.DataFrame(lead_types_produced.groupby(cols)['
    Quantity'].sum())
3 lead_types_scrapped_group = pd.DataFrame(lead_types_scrapped.groupby(cols)['
    Quantity'].sum())
4 percentage = lead_types_produced_group.join(lead_types_scrapped_group, on=['
    Material', 'Year', 'Month'], lsuffix='_produced', rsuffix='_scrapped')
5 percentage['%'] = percentage['Quantity_scrapped']/percentage['Quantity_produced'
    ]
6 percentage = percentage.reset_index()
7 percentage = percentage[percentage['%'].notnull()]
```

Predicción del porcentaje de desperdicio

Se aplica una regresión lineal para cada número de parte con el fin de estimar el porcentaje de scrap en el mes siguiente:

```

1 from sklearn.linear_model import LinearRegression
2
3 predictions = []
4
5 for i in tenLeadTypes:
6     y = np.array(percentage[percentage['Material']==i]['%'])
7     X = np.array(range(0, len(y)))
8     X = X.reshape(-1,1)
9     model = LinearRegression()
10    model.fit(X, y)
11    X_predict = [[len(X)+1]] # put the dates of which you want to predict kwh
                             here
12    y_predict = model.predict(X_predict)
13    print(i, y, y_predict)
14    predictions.append(y_predict[0])
15
16 values = pd.DataFrame([tenLeadTypes, predictions])

```

Cálculo de producción ajustada

Finalmente, se ajustan las predicciones de ventas para incluir el desperdicio estimado. El modelo GRU original predice las unidades necesarias para venta, y con base en la proyección del *scrap* se calcula la cantidad que debe ser efectivamente fabricada:

```

1 scrap_predicted=pd.DataFrame(columns=['unique_id', 'percentage'])
2 scrap_predicted['unique_id'],scrap_predicted['percentage'] = values.transpose()
   [0],values.transpose()[1].astype('float')+1
3
4 predictions_2024 = pd.read_csv('predicts_2024.csv')
5 predictions_2024.ds=pd.to_datetime(predictions_2024['ds'], format='%Y-%m-%d')
6 predictions_2024['Year']=predictions_2024.ds.dt.year
7 predictions_2024['Month']=predictions_2024.ds.dt.month

```

```
8 predictions_2024['Week']=predictions_2024.ds.dt.isocalendar().week
9 predictions_2024['DayofWeek']=predictions_2024.ds.dt.dayofweek
10
11 predicts_2024_scrap = predictions_2024.merge(scrap_predicted,how='inner',on=[
    unique_id'])
12 predicts_2024_scrap['GRU_new']=predicts_2024_scrap['GRU']*predicts_2024_scrap['
    percentage']
```

Este proceso permite obtener una estimación realista de la cantidad de unidades que deben fabricarse mensualmente, integrando predicción de ventas, proyecciones de desperdicio y el comportamiento histórico del sistema productivo. Esta lógica será utilizada como base para futuras aplicaciones de planeación de producción automatizada.

3.9. Generar un modelo para la predicción de producción semanal

En esta etapa del proyecto, ya se cuenta con una estimación de la cantidad de piezas que deben ser fabricadas para satisfacer la demanda proyectada, la cual ha sido ajustada para considerar el desperdicio estimado (*scrap*). Sin embargo, para implementar dicha producción en un entorno real, es indispensable contemplar las restricciones operativas propias del sistema de manufactura, tales como la disponibilidad de tiempo por turno, el número de líneas de producción activas y la capacidad efectiva de las máquinas.

Ante este contexto, se propone el desarrollo de un modelo que permita distribuir de manera eficiente la carga de trabajo a lo largo de cada semana. El objetivo es generar una planificación semanal de producción que sea factible, balanceada y coherente con los recursos disponibles, asegurando que la meta de producción ajustada pueda cumplirse sin sobrecargar las líneas ni poner en riesgo los compromisos de entrega.

Como paso previo, es necesario consolidar la predicción mensual en un esquema semanal que refleje adecuadamente la estructura operativa. Para ello, se agrupan

3.9. GENERAR UN MODELO PARA LA PREDICCIÓN DE PRODUCCIÓN SEMANAL

los registros por semana, número de parte y año, calculando la suma de unidades requeridas, tanto desde la perspectiva del volumen original de ventas como del volumen ajustado por *scrap*.

```
1 predicts_2024_scrap = predicts_2024_scrap.groupby(  
2     ['Year', 'Month', 'Week', 'unique_id']  
3 ).agg({  
4     'y': 'sum',  
5     'GRU_new': 'sum'  
6 }).reset_index()
```

Capítulo 4

Resultados y discusión

4.1. Resultados

4.1.1. Obtención de datos

Durante la recolección y procesamiento de los datos históricos de ventas, se verificó que no existen registros con valores negativos. Las ventas negativas, comúnmente asociadas a procesos de devolución de producto, no fueron consideradas en este análisis, ya que pertenecen a una operación distinta y no reflejan directamente el comportamiento de la demanda real. Por tanto, los datos utilizados corresponden exclusivamente a ventas efectivas, lo cual permite mantener la coherencia del modelo de predicción basado en consumos reales.

4.1.2. Factibilidad de modelo GRU para la predicción de ventas

El primer paso en la experimentación consistió en evaluar la factibilidad de aplicar un modelo basado en una Unidad Recurrente Cerrada (GRU) para la predicción de ventas.

Para esta evaluación inicial, se utilizó un conjunto de entrenamiento que abarca el periodo comprendido entre los años 2018 y 2022. El conjunto de validación corres-

ponde al año 2023, lo que permite evaluar el modelo frente a datos recientes no vistos durante el entrenamiento.

En este primer acercamiento, se agrupan todas las ventas sin segmentación por número de parte, con el objetivo de simplificar el modelo y obtener una visión general de su comportamiento. Los días sin ventas fueron conservados con valor cero, asegurando así la continuidad temporal de la serie.

Los resultados obtenidos se presentan en la Tabla 4.1. A nivel anual, la diferencia entre las ventas reales y las predichas es de tan solo un 2 %, lo cual resulta prometedor. Sin embargo, el Error Absoluto Medio (MAE) es considerablemente alto, con una diferencia promedio diaria de casi 26 mil piezas, lo que sugiere que el modelo aún no captura correctamente las fluctuaciones diarias.

Tabla 4.1: Resultados obtenidos de la predicción del modelo GRU en las ventas sumarizadas.

Parametro	Resultado (pcs)
Ventas Reales 2023	14.1M
Ventas Predichas 2023	14.3 M
MAE	25 989

A pesar del alto error absoluto diario, el resultado acumulado anual es bastante preciso. De hecho, al compararlo con el pronóstico realizado por un analista experto quien estimó ventas de 14.26 millones de piezas para 2023 se observa una gran similitud, lo cual valida el potencial del modelo GRU. Este resultado inicial justifica continuar con la optimización de hiperparámetros y segmentación por número de parte, con el fin de aumentar la precisión a nivel operativo.

La Figura 4.1 muestra la evolución temporal de las ventas reales y su comparación con las predicciones generadas por el modelo GRU.

En cuanto al rendimiento computacional, el entrenamiento de este modelo en CPU tomó aproximadamente 1 hora con 15 minutos. Al migrar la ejecución a una GPU, el tiempo se redujo significativamente a tan solo 9 minutos, lo que demuestra la viabilidad de escalar este enfoque mediante aceleración por hardware.

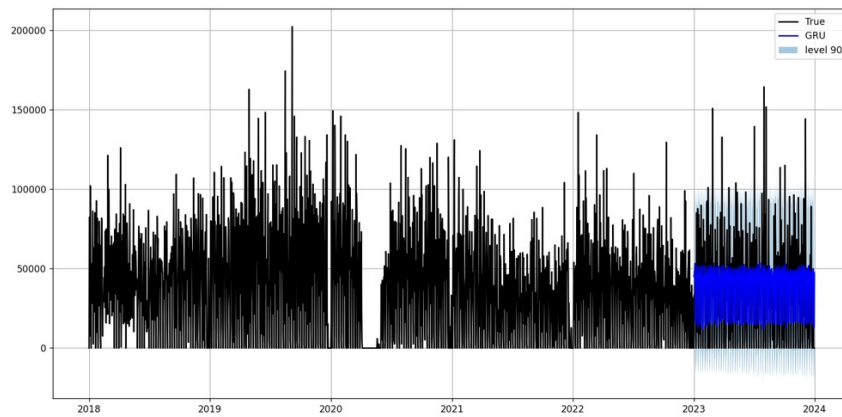


Figura 4.1: Comparación entre ventas reales y predicción generada por el modelo GRU (ventas agregadas).

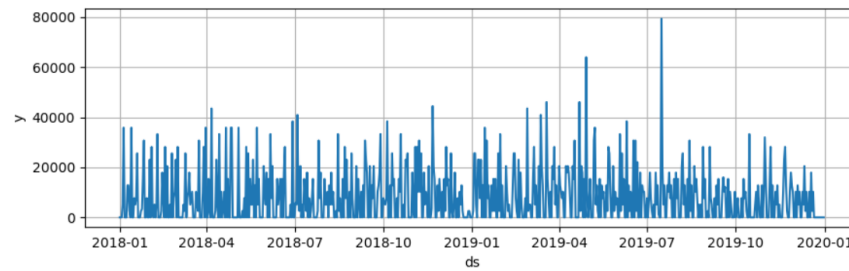
4.1.3. Predicción del periodo en pandemia

Con el objetivo de mejorar la precisión del modelo, se decidió entrenar una red GRU de forma individual para cada número de parte, ya que el comportamiento de ventas varía significativamente entre productos. Esta decisión se basa en la observación empírica de los datos, donde se identifican patrones de venta heterogéneos entre diferentes sensores de presión.

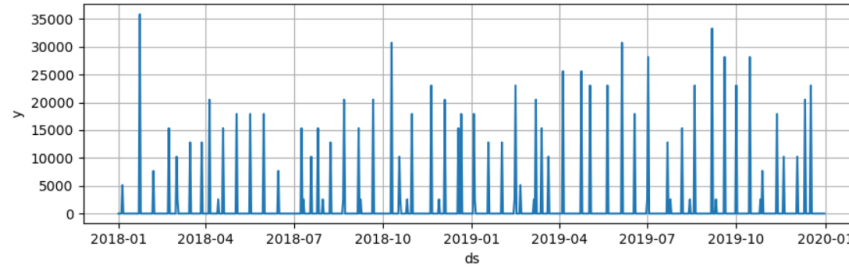
En la Figura 4.2 se ilustran dos casos contrastantes, en la subfigura 4.2a, se observa un patrón de ventas casi aleatorio, mientras que en la subfigura 4.2b se evidencia un comportamiento periódico, donde las ventas se concentran de forma regular en un día específico de la semana. Estos ejemplos justifican la necesidad de una modelación diferenciada por producto.

Debido a las limitaciones computacionales asociadas al proceso de ajuste de hiperparámetros mediante *Grid Search* para cada modelo individual, se optó por reducir el número de productos a modelar. En lugar de procesar los 45 números de parte disponibles en la base de datos, se seleccionaron los 10 con mayor volumen de ventas durante el año 2023, lo cual permite enfocar los recursos en los productos de mayor impacto comercial.

Adicionalmente, se decidió incluir el número de parte 368, que dejó de venderse en ese mismo año y no presenta un patrón de venta regular. Su inclusión tiene como



(a) Venta casi aleatoria de un número de parte X



(b) Venta semanal en los días jueves de un número de parte Y

Figura 4.2: Comportamiento de venta acorde a número de parte

propósito introducir un caso atípico o ruido dentro del conjunto de entrenamiento, con el objetivo de evaluar cómo este tipo de anomalías puede afectar el desempeño general del modelo de predicción.

La Tabla 4.2 muestra la lista de los diez números de parte seleccionados junto con su respectiva participación en las ventas del año 2023.

Tabla 4.2: Los diez números de parte que más se venden.

Número de Parte	Participación de ventas año 2023
162	16.3%
213	12.7%
339	3.8%
459	9.9%
472	4.0%
527	6.5%
572	6.2%
319	5.9%
329	6.8%
368	0.0%
Resto	27.9%

Esta selección permitirá evaluar la capacidad del modelo GRU para adaptarse a com-

portamientos heterogéneos, tanto en productos de alta rotación como en aquellos con ventas interrumpidas o no recurrentes. La selección de los diez números de parte más representativos, se abarca el 72.1% del total de ventas registradas en 2023, lo que equivale a aproximadamente 10.2 millones de piezas. Este subconjunto es considerado suficientemente representativo para realizar la experimentación y evaluación del modelo, manteniendo un balance adecuado entre cobertura y eficiencia computacional.

Para el ajuste de hiperparámetros se implementó una búsqueda por malla (Grid Search) sobre el espacio definido en los parámetros clave del modelo GRU y su capa de salida tipo MLP. La configuración de la malla considera las siguientes combinaciones:

- Épocas:[500,1000,2000]
- Capas de entrada en GRU:[2]
- Neuronas de salida en GRU:[512]
- Contexto:[1]
- Poda:[0.5]
- Neuronas en MLP:[64,128]
- Capas en MLP:[2]
- Tasa de aprendizaje:[0.01]
- Normalización:['robust']

Esta configuración da lugar a 6 combinaciones distintas de hiperparámetros, que serán evaluadas individualmente para cada número de parte. El objetivo es identificar la combinación que optimice el rendimiento del modelo según las métricas de error seleccionadas, particularmente MAE y YoY.

Como se observa en la Figura 4.1, el comportamiento de ventas presenta valores atípicos durante el periodo de pandemia por COVID-19, específicamente entre los

meses de marzo y mayo de 2020. Dado que estos valores no responden a un patrón de comportamiento comercial ordinario, se plantea realizar una predicción específica para dicho periodo y analizar si sustituir estos datos mejora el rendimiento general del modelo.

Para ello, se definió un nuevo conjunto de entrenamiento que abarca el periodo de enero de 2018 a diciembre de 2019, mientras que el conjunto de validación incluye los meses de enero y febrero de 2020. Esta separación permite evaluar el modelo antes del impacto de la pandemia, para posteriormente utilizarlo en la generación de predicciones durante el periodo anómalo.

Se aplicó una búsqueda exhaustiva de combinaciones de hiperparámetros utilizando *Grid Search* para cada número de parte, registrando los resultados en un archivo de Excel. Para cada combinación evaluada, se calcularon las métricas correspondientes y se seleccionó el modelo con el menor Error Absoluto Medio (MAE). En la Figura 4.3 se muestra un ejemplo del resultado obtenido para el número de parte 162 con los datos originales del periodo de pandemia.

	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
PN	h	max_steps	encoder_n_layers	encoder_hidden_size	context_size	encoder_dropout	decoder_hidden_size	decoder_layers	learning_rate	scaler_type	Year	MAE	R2	Pearson	Best Model
162	60	500	2	512	1	0.5	64	2	0.01	robust	1.273674893	5158.604509	0.172889220	0.51337079	0
162	60	500	2	512	1	0.5	128	2	0.01	robust	1.230785700	4430.5682	0.450032712	0.714751337	1
162	60	1000	2	512	1	0.5	64	2	0.01	robust	1.369643239	5330.035466	-0.01688277	0.458143741	0
162	60	1000	2	512	1	0.5	128	2	0.01	robust	1.174178592	4667.110762	0.356155756	0.62279404	0
162	60	2000	2	512	1	0.5	64	2	0.01	robust	1.365732844	5628.201559	0.253725875	0.61946521	0
162	60	2000	2	512	1	0.5	128	2	0.01	robust	1.238126897	4898.811392	0.322499058	0.617341626	0

Figura 4.3: *Grid Search* aplicado al número de parte 162 utilizando datos originales del periodo de pandemia.

Para evaluar el desempeño de los modelos durante el proceso de validación, se utilizaron las siguientes métricas: el Error Absoluto Medio (MAE), definido en la ecuación (Ecuación 2.25); el coeficiente de determinación R^2 , presentado en la ecuación (Ecuación 2.10); y la variación interanual (Year-over-Year, YoY), descrita en la ecuación (Ecuación 2.11). Estas métricas permiten cuantificar la precisión de las predicciones tanto a nivel puntual como en términos de tendencia general.

Los resultados obtenidos para cada uno de los productos modelados con la arquitectura GRU se presentan en la Tabla 4.3. En ella se reportan las métricas de evaluación correspondientes al conjunto de validación: el Error Absoluto Medio (MAE), el coeficiente de determinación R^2 y la métrica de variación interanual YoY, todos expresados

como porcentaje cuando corresponde.

Estas métricas permiten analizar tanto la precisión diaria de las predicciones como su capacidad para seguir la tendencia global de ventas. Es importante destacar que algunos productos, como el número de parte 572 y 368, muestran un desempeño notablemente superior, con valores elevados de R^2 , mientras que otros, como el producto 339 y 319, presentan dificultades para ser modelados con precisión, reflejadas en valores negativos de R^2 .

Tabla 4.3: Métricas obtenidas de la predicción con el modelo GRU.

Producto	MAE_original	R2_original%	YoY_original%
162	4431	45	123
213	7572	21	101
339	3132	-6	40
459	3420	27	62
472	1622	46	50
527	3948	16	81
572	185	95	101
319	3774	-26	100
329	2062	40	70
368	1190	92	105

Con el fin de mejorar el proceso de ajuste de hiperparámetros y la capacidad de generalización del modelo, se plantea una estrategia de tratamiento para los valores atípicos presentes en los datos de entrenamiento. Esta estrategia consiste en reemplazar los valores atípicos por el valor máximo registrado en su respectivo año, con el objetivo de reducir la variabilidad extrema que puede dificultar el entrenamiento de redes neuronales recurrentes.

La Figura 4.4 ilustra el procedimiento aplicado para fijar los valores atípicos. Esta transformación busca suavizar los picos inusuales en las series temporales, evitando que el modelo sobreajuste a eventos poco representativos del comportamiento general.

Este ajuste permite generar un nuevo conjunto de entrenamiento con menor dispersión extrema, lo que puede favorecer la estabilidad del modelo durante la optimización y mejorar su rendimiento en la predicción de tendencias generales.

La Figura 4.5 muestra una comparación entre los datos originales y los datos modi-

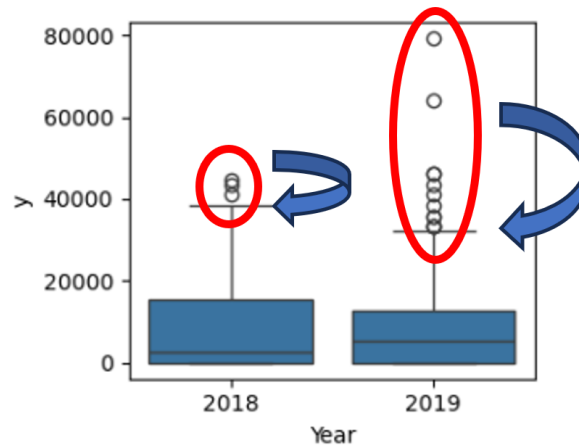
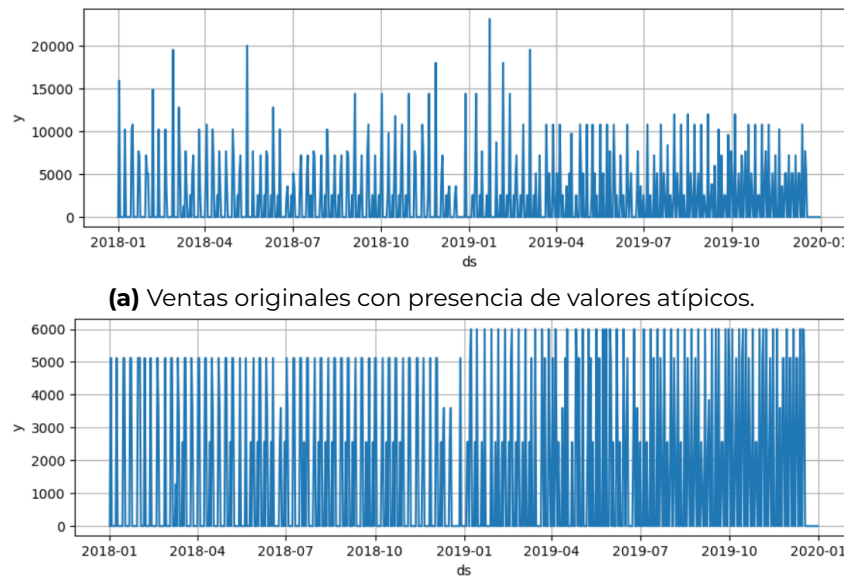


Figura 4.4: Reemplazo de valores atípicos por el valor máximo del año correspondiente.

ficados mediante esta técnica. En la subfigura 4.5a se observan los valores atípicos extremos, mientras que en la subfigura 4.5b se presenta la misma serie tras el ajuste, con los picos suavizados al nivel máximo anual.



(a) Ventas originales con presencia de valores atípicos.

(b) Ventas ajustadas con valores atípicos reemplazados por el valor máximo anual.

Figura 4.5: Comparación entre datos originales y datos con valores atípicos fijados.

Este procedimiento se evaluará empíricamente para determinar si conduce a mejoras en las métricas de desempeño del modelo, particularmente en términos de MAE, R^2 y YoY.

Los resultados obtenidos tras aplicar el tratamiento de fijar los valores atípicos al valor máximo anual se presentan en la Tabla 4.4. Al comparar estas métricas con las reportadas previamente en la Tabla 4.3, se observa que, contrariamente a lo esperado, no se obtuvo una mejora en el desempeño del modelo. De hecho, en varios casos se evidenció una disminución en los valores de R^2 y del coeficiente de correlación de Pearson, con algunos productos presentando incluso valores NaN en esta última métrica.

Este comportamiento puede explicarse por la naturaleza de ciertos productos cuya venta ocurre en intervalos semanales específicos. En estos casos, la imputación de ceros en los días sin venta altera significativamente la distribución original de la serie, desplazando artificialmente la tendencia hacia el valor cero. Como resultado, el modelo no logra identificar correctamente los patrones reales de comportamiento.

La Figura 4.6 ilustra esta situación mediante un diagrama de caja, donde se aprecia que el bigote superior está en cero y que los valores reales de venta son interpretados como atípicos por el sistema, al encontrarse fuera del rango intercuartílico. Esta distorsión en la distribución de los datos compromete la capacidad del modelo para generalizar de forma efectiva.

Tabla 4.4: Resultados del modelo GRU con valores atípicos fijados al máximo anual.

Producto	MAE_atípico	R2_atípico%	YoY_atípico%	Pearson_atípico%
162	4998	33	127	64
213	7417	25	100	53
339	2389	-11	0	nan
459	3511	25	70	56
472	2688	29	0	nan
527	3900	5	51	37
572	984	-17	0	nan
319	3887	-18	89	10
329	2250	18	38	79
368	3467	-17	0	nan

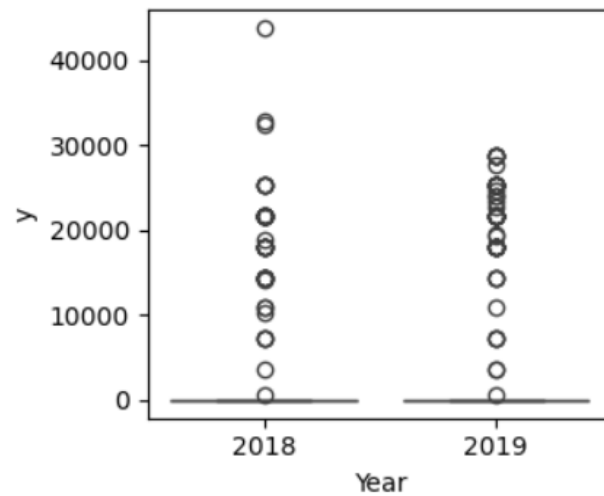


Figura 4.6: Diagrama de caja de un producto con venta semanal. El modelo interpreta los valores reales como atípicos.

4.1.4. Predicción del periodo de pandemia con datos originales

Tras comprobar que el reemplazo de los valores atípicos por el valor máximo anual no mejora e incluso deteriora el rendimiento del modelo, se optó por mantener el conjunto de datos original y realizar una búsqueda más exhaustiva de hiperparámetros con el objetivo de optimizar el entrenamiento.

La malla definida para la búsqueda por *Grid Search* incluyó una mayor variedad de combinaciones relevantes para la arquitectura GRU y el modelo de salida MLP. La configuración evaluada se detalla a continuación:

- Épocas:[5120,1024,512]
- Función de activación de GRU:['relu']
- Capas de entrada en GRU:[2]
- Neuronas de salida en GRU:[512]
- Contexto:[1]
- Poda:[0.5]
- Neuronas en MLP:[128]

- Capas en MLP:[2]
- Tasa de aprendizaje:[0.001]
- Normalización:['robust']
- Funciones de pérdida en el aprendizaje:[Poisson, MAE, MSE, RMSE]

Debido a la extensión de la malla, el tiempo promedio de búsqueda por número de parte fue de aproximadamente 2.5 horas, lo cual se consideró aceptable dadas las capacidades computacionales disponibles y la importancia de encontrar una configuración óptima.

El conjunto de entrenamiento se estableció desde enero de 2018 hasta diciembre de 2019, y el conjunto de validación incluyó los meses de enero y febrero de 2020, es decir, justo antes del inicio del periodo de pandemia. Los resultados obtenidos se presentan en la Tabla 4.5..

Tabla 4.5: Resultados del modelo GRU con conjunto de datos original y búsqueda de hiperparámetros ampliada.

Producto	MAE	R2 %	YoY%	Pearson %
162	4287	48	116	72
213	5608	49	98	72
339	2215	41	80	65
459	3349	34	72	61
472	1530	49	59	74
527	4119	-20	91	28
572	254	94	100	97
319	2153	57	101	75
329	1540	56	66	82
368	954	92	82	98

Al comparar estos resultados con los reportados previamente en la Tabla 4.3, se observa una mejora generalizada en el desempeño del modelo, especialmente en las métricas R^2 y Pearson, lo cual indica una mejor capacidad de ajuste y correlación con los datos reales. La única excepción significativa es el producto 527, que presenta un descenso en ambas métricas, posiblemente debido a una mayor variabilidad en su comportamiento.

Con los modelos optimizados se procedió a realizar la predicción del periodo correspondiente a la pandemia (marzo a mayo de 2020). Cabe señalar que no existen métricas de validación directa para este periodo, ya que las ventas reales se detuvieron por completo; sin embargo, el modelo permite estimar lo que habría sido un comportamiento esperable en condiciones normales. La Figura 4.7 muestra un ejemplo representativo de la predicción generada para este intervalo.

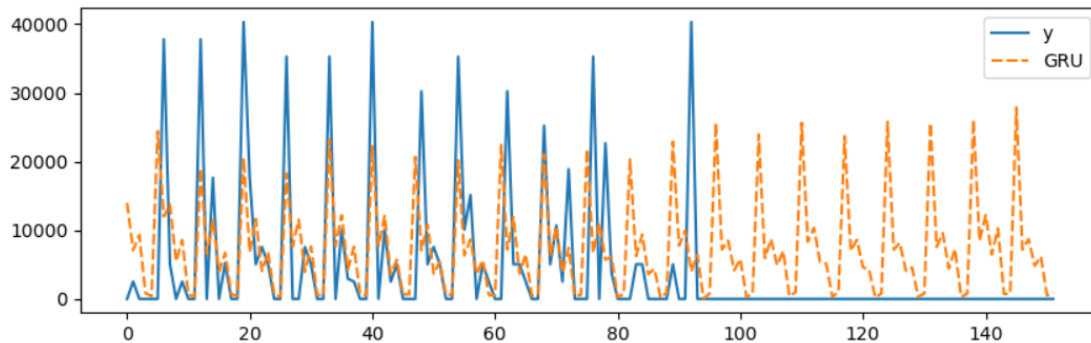


Figura 4.7: Predicción del modelo GRU para el periodo de pandemia (marzo a mayo de 2020).

Estos resultados permiten contar con una estimación robusta para reemplazar el segmento afectado por el COVID-19, lo cual resulta útil para alimentar futuros modelos de planeación y análisis de escenarios.

4.1.5. Predicción del año 2023 con datos originales y periodo de pandemia cambiado

Con el periodo de pandemia (marzo a mayo de 2020) ahora sustituido por los valores predichos mediante el modelo GRU, se procede a evaluar el impacto de esta modificación sobre el rendimiento general del modelo al predecir las ventas del año 2023. El objetivo es comparar el desempeño del modelo utilizando el conjunto de datos original frente a aquel que contiene los valores imputados, manteniendo el mismo marco temporal para el entrenamiento y validación.

Para ambos enfoques, el conjunto de entrenamiento abarca el periodo comprendido entre enero de 2018 y diciembre de 2022, mientras que el conjunto de validación corresponde al año 2023.

La búsqueda de hiperparámetros se llevó a cabo mediante *Grid Search*, utilizando la siguiente configuración:

- Épocas:[5120,1024,512]
- Función de activación de GRU:['relu']
- Capas de entrada en GRU:[4,2]
- Neuronas de salida en GRU:[512]
- Contexto:[1]
- Poda:[0.5]
- Neuronas en MLP:[128]
- Capas en MLP:[2]
- Tasa de aprendizaje:[0.001]
- Normalización:['robust']
- Funciones de pérdida en el aprendizaje:[Poisson, MAE, MSE, RMSE]

En esta iteración, se amplió el espacio de búsqueda añadiendo una segunda opción para el número de capas GRU (2 y 4), lo cual duplicó el tiempo computacional promedio por producto, pasando de 2.5 a aproximadamente 5 horas por Grid Search.

El propósito de esta comparación es determinar si la sustitución de los datos nulos del periodo de pandemia con predicciones realistas contribuye a mejorar la capacidad del modelo para prever el comportamiento de ventas en un año completo como 2023. La Figura 4.8 muestra un ejemplo visual de la diferencia entre ambos enfoques.

Este análisis permitirá evaluar empíricamente si el tratamiento del periodo anómalo mejora la generalización del modelo y reduce los errores de predicción en escenarios posteriores, como el comportamiento pospandemia.

Los resultados obtenidos para la predicción del año 2023 utilizando los datos originales se muestran en la Tabla 4.6, mientras que la Tabla 4.7 presenta los resultados

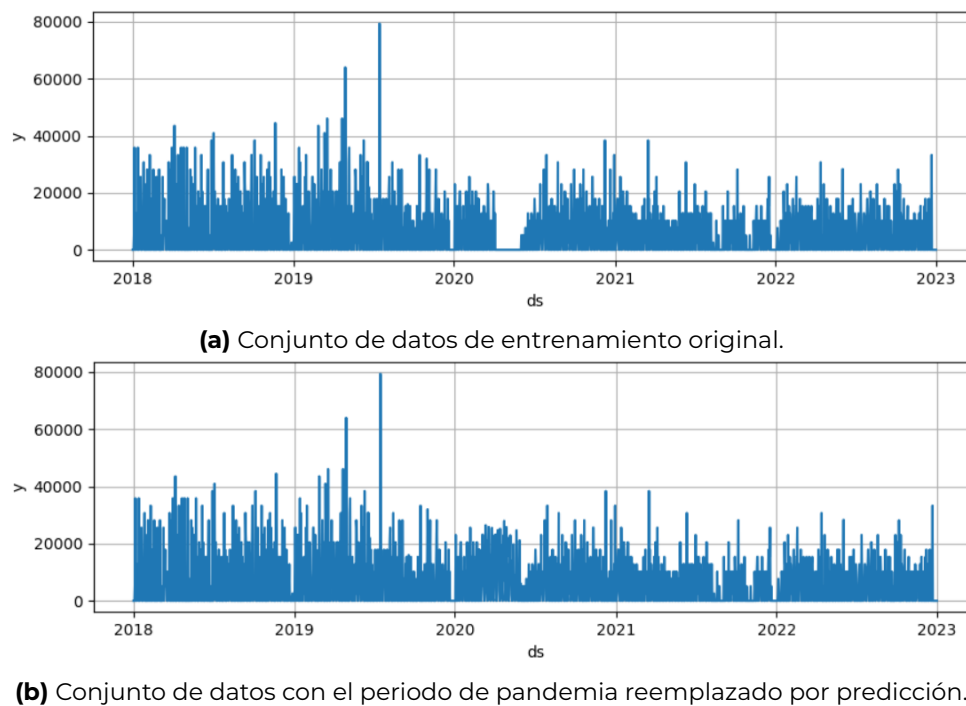


Figura 4.8: Comparación entre conjuntos de datos con y sin modificación del periodo de pandemia

utilizando el conjunto de datos con el periodo de pandemia (marzo a mayo de 2020) reemplazado por valores generados por el modelo GRU.

Al comparar ambas tablas, se observa que la sustitución del periodo de pandemia con valores predichos no conlleva a una mejora significativa en las métricas de evaluación. En algunos casos, incluso se presenta una degradación en el desempeño, como en los productos 162, 572 y 368. Este último, en particular, genera valores fuera de rango debido a que fue descontinuado en 2023, lo que justifica su comportamiento anómalo.

Además, esta estrategia implica duplicar el tiempo de entrenamiento, ya que se requiere un modelo inicial para predecir el periodo de pandemia y posteriormente un segundo modelo para proyectar las ventas de 2023. En consecuencia, se concluye que el costo computacional adicional no se ve reflejado en un beneficio sustancial en las predicciones.

Estos resultados también abren la posibilidad de replantear la extensión temporal

Tabla 4.6: Predicción del modelo GRU para el año 2023 Conjunto de datos original.

Producto	MAE	R2 %	YoY%	Pearson%
162	4530	17	95	46
213	4975	20	98	46
339	1480	-8	-24	5
459	3527	31	128	60
472	2108	-2	51	15
527	3291	10	103	32
572	2363	-7	14	14
319	1863	-4	67	44
329	2232	2	73	48
368	432	-300	4906	0

Tabla 4.7: Predicción del modelo GRU para el año 2023 Periodo de pandemia sustituido.

Producto	MAE	R2 %	YoY%	Pearson%
162	4556	8	77	42
213	5039	17	108	45
339	2618	2	103	16
459	3387	33	118	60
472	2313	-7	71	10
527	3248	2	102	28
572	2485	6	68	27
319	1981	-14	88	49
329	2014	18	71	52
368	450	-537	6947	-3

del conjunto de entrenamiento. Dado que las redes neuronales recurrentes tienden a dar mayor peso a los datos más recientes (debido a su estructura secuencial), es válido cuestionar si incluir datos tan lejanos como los del año 2018 resulta realmente beneficioso. Se propone, por tanto, una línea de trabajo futura que explore modelos entrenados únicamente con información desde 2021 en adelante, lo cual podría reducir significativamente el tiempo de entrenamiento sin comprometer la capacidad predictiva.

4.1.6. Predicción de ventas para el año 2023 con un conjunto de entrenamiento menor

En esta etapa del análisis, se evalúa el impacto de utilizar un conjunto de entrenamiento más reciente, limitado únicamente al periodo 2021-2022, mientras que el conjunto de validación permanece como el año completo 2023. Esta configuración busca explorar si el uso exclusivo de datos posteriores al periodo de pandemia es suficiente para lograr predicciones precisas, considerando que las redes neuronales recurrentes tienden a favorecer patrones recientes debido a su estructura secuencial.

Los resultados obtenidos se presentan en la Tabla 4.8.

Tabla 4.8: Predicción del modelo GRU para el año 2023 Entrenamiento con datos de 2021-2022.

Producto	MAE	R2 %	YoY%	Pearson%
162	4695	17	94	44
213	5162	17	98	42
339	2969	-97	132	14
459	3471	27	115	58
472	2562	-20	88	15
527	3358	3	102	24
572	2431	9	52	36
319	1888	22	123	57
329	2026	26	69	55
368	478	-440	7460	-2

Al comparar estos resultados con los obtenidos previamente en la Tabla 4.6, se observa que no existen diferencias sustanciales en el rendimiento del modelo al utilizar un

conjunto de entrenamiento más corto. Las métricas como MAE, R^2 y Pearson se mantienen dentro de rangos similares, e incluso algunos productos presentan ligeras mejoras en ciertos indicadores.

Este comportamiento puede explicarse por dos razones fundamentales:

- El contexto de la arquitectura GRU se encuentra fijado en 1, lo que implica que la predicción de cada valor se basa únicamente en el valor anterior. Esta configuración limita la influencia de datos históricos lejanos, haciendo que el modelo se enfoque en los patrones inmediatos.
- La naturaleza secuencial y la poda (dropout) aplicada durante el entrenamiento promueven la desatención a patrones antiguos, priorizando información reciente para evitar sobreajuste.

Además de mantener una precisión comparable, la reducción del conjunto de entrenamiento tuvo un impacto positivo en el tiempo de cómputo, disminuyendo en aproximadamente un 50% la duración total del proceso de entrenamiento por modelo. Este ahorro computacional es significativo, especialmente si se contempla la implementación del sistema a nivel planta para múltiples líneas de producción y productos.

En conclusión, limitar el conjunto de entrenamiento al periodo pospandemia (2021-2022) es una alternativa viable que reduce los recursos computacionales sin comprometer significativamente el rendimiento predictivo del modelo.

4.1.7. Análisis del cambio en la generación de *scrap* ante variaciones en la línea de producción

Con el objetivo de identificar posibles patrones en la generación de *scrap* asociados a cambios en la línea de producción, se llevó a cabo un análisis exploratorio utilizando la herramienta de visualización Power BI. Esta herramienta permitió visualizar el comportamiento de producción y desperdicio por número de parte y línea de ensamblaje, buscando detectar variaciones significativas al cambiar de línea.

Sin embargo, tras el análisis de los datos históricos, se observó que no existen registros con variación en la línea de producción para un mismo número de parte. Específicamente, cada producto se fabrica de forma exclusiva en una sola línea, sin rotación entre líneas a lo largo del tiempo, como se puede apreciar en la Figura 4.9, donde se muestra el caso del número de parte 339.

Producción por línea del 2018 al 2023							
# Producto	3	4	5	6	7	8	Total
339							
Piezas scrap	372	83,835	0				84,207
Piezas producidas	0	3,214,894	0	0	0	0	3,214,894
%		2.61 %					2.62 %
Piezas scrap	372	83,835	0				84,207
Piezas producidas	0	3,214,894	0	0	0	0	3,214,894
%		2.61 %					2.62 %

Figura 4.9: Generación de *scrap* y producción del número de parte 339, se observa que no hay variación de líneas de producción (columnas) para realizar los análisis.

Para complementar este hallazgo, se realizó una entrevista con el equipo de gerencia de producción, donde se confirmó que las líneas no se modifican para productos de alta rotación. El motivo principal es operativo: cambiar un producto de línea implica detener la línea por completo, lo que representa una pérdida estimada de al menos 6 horas de producción, debido al tiempo requerido para el vaciado, limpieza y reconfiguración del equipo.

Como consecuencia de esta política operativa, no existen datos que permitan evaluar el impacto del cambio de línea sobre la generación de *scrap*, ya que dicha variación simplemente no ocurre en la práctica bajo las condiciones actuales de la planta.

En conclusión, este análisis no pudo ser desarrollado debido a la falta de variabilidad en los datos y a restricciones operativas inherentes al proceso de producción. No obstante, esta observación permite entender mejor las limitaciones del entorno productivo y representa un insumo valioso para futuros rediseños de procesos que busquen mayor flexibilidad en la asignación de líneas.

4.1.8. Generar un modelo para la predicción de producción semanal

Hasta este punto, la predicción de demanda se ha realizado considerando únicamente las ventas efectivas. No obstante, para una planificación de producción más realista, es indispensable incorporar las pérdidas ocasionadas por la generación de *scrap*, ya que estas piezas deben ser fabricadas pero no llegan a comercializarse.

Para ello, se estimó la tasa de *scrap* promedio tomando como base los datos históricos del año 2023. En particular, se calculó el promedio mensual de desperdicio para cada número de parte, y se utilizó este valor como factor de corrección en la predicción de producción para el año 2024. La Figura 4.10 muestra el caso del producto 162, donde se observa la evolución del *scrap* durante el año y el promedio empleado en la predicción.

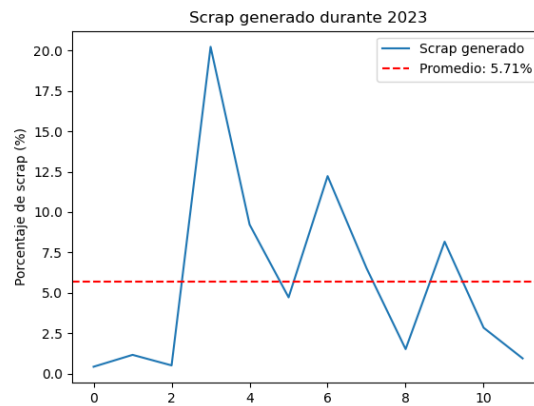


Figura 4.10: Generación mensual de *scrap* en 2023 para el producto 162 y su promedio utilizado para la predicción.

El valor promedio de *scrap* fue sumado a la predicción de ventas para obtener una estimación más precisa de la producción requerida. La Figura 4.11 presenta un ejemplo de esta integración, mostrando la curva de predicción ajustada que incluye el volumen adicional necesario para compensar el desperdicio.

A partir de este procedimiento, se obtuvo la predicción de producción semanal para cada uno de los productos analizados. Los resultados se resumen en la Tabla 4.9, en la que se reportan las métricas de desempeño para cada número de parte, destacando

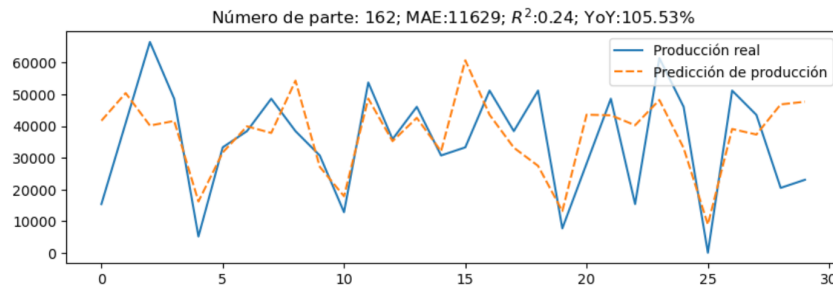


Figura 4.11: Predicción de producción total considerando la tasa de generación de *scrap*.

el impacto del ajuste por *scrap*.

Tabla 4.9: Predicción del modelo GRU ajustada por *scrap* con entrenamiento del periodo 2021 a 2022.

Producto	MAE	R2 %	YoY %
162	11629	24	105.53
213	12743	6	107.65
339	13491	-27	130.81
459	13457	-17	157.37
472	8518	-22	79.41
527	8909	31	133.56
572	6674	-22	73.83
319	14023	-60	233.97
329	7132	-59	67.47
368	2874	-265	4750.65

En general, los valores de YoY aumentan significativamente, como era de esperarse al incorporar el efecto del *scrap* a las predicciones. Sin embargo, también se observa un aumento en el MAE y deterioro del R^2 en varios productos, lo que indica que la variabilidad adicional introducida por el *scrap* representa un desafío para la estabilidad del modelo. Estos resultados indican que, si bien es necesario ajustar la predicción de producción con base en el desperdicio, también podría explorarse el uso de técnicas complementarias para modelar el *scrap* de forma separada o por categorías de producto.

4.2. Discusión

Los resultados obtenidos en este estudio permiten concluir que el modelo propuesto basado en una red neuronal recurrente tipo GRU logra realizar predicciones de ventas ajustadas a los requerimientos establecidos para esta tesis, cumpliendo con el objetivo central de desarrollar una metodología automatizada de pronóstico que reemplace, complemente o mejore el proceso actual basado en juicio experto.

Sin embargo, es importante señalar que la validación completa del modelo con respecto al año 2024 aún no puede realizarse en su totalidad, ya que al momento de la redacción de este documento no se dispone del conjunto completo de datos de ventas para ese periodo. Por lo tanto, los análisis y métricas presentadas corresponden únicamente al primer semestre de 2024 o, en su defecto, se han utilizado datos históricos hasta 2023 para simular condiciones reales.

Dado lo anterior, se ha optado por utilizar como métrica principal de evaluación la variación interanual (YoY). Esta métrica permite una comparación directa con las proyecciones actualmente realizadas por especialistas en planeación, quienes suelen basar sus estimaciones en la tendencia de ventas del año anterior y en factores de mercado externos. La ventaja de esta métrica radica en que, al expresar la predicción como una proporción respecto al año previo, se facilita la interpretación del modelo por parte del personal operativo, directivo y analítico de la empresa.

No obstante, al revisar las demás métricas cuantitativas como el Error Absoluto Medio (MAE) y el coeficiente de determinación R^2 , se observa que el desempeño del modelo es modesto en comparación con los resultados reportados en estudios de series temporales más tradicionales. Esta diferencia puede atribuirse a varios factores:

- La naturaleza altamente variable e irregular de la demanda en ciertos productos.
- La fijación del contexto de la GRU en un valor mínimo de 1, lo cual limita la memoria a corto plazo y reduce la capacidad del modelo para capturar patrones más complejos.

- La inclusión del *scrap* como valor añadido en la predicción de producción, lo cual introduce ruido adicional que incrementa la variabilidad de los errores.

A pesar de ello, el modelo cumple su función dentro del marco de esta tesis, demostrando que es posible automatizar el pronóstico de ventas y producción con un nivel de precisión comparable al proceso humano actual, y además lograr una reducción significativa en tiempo de análisis y recursos requeridos para su ejecución.

En aplicaciones reales, se sugiere complementar el modelo con mecanismos de retroalimentación periódica conforme se disponga de nuevos datos, así como explorar estrategias de ajuste dinámico del contexto en la GRU para capturar relaciones de dependencia más profundas. Asimismo, las métricas MAE y R^2 deben considerarse con cautela como criterios secundarios, dado que en entornos industriales la comparación directa año a año (YoY) sigue siendo el estándar operativo.

Capítulo 5

Conclusiones y recomendaciones

5.1. Conclusiones

Este estudio propuso la aplicación de redes neuronales recurrentes tipo GRU para la predicción de ventas en el contexto de la industria automotriz, específicamente para sensores de presión fabricados por una empresa privada. Los datos utilizados abarcaron el periodo comprendido entre enero de 2018 y julio de 2024, con especial atención al tratamiento de valores atípicos generados durante la pandemia por COVID-19 y al impacto del *scrap* en la planeación de la producción.

A continuación, se presentan las principales conclusiones derivadas del trabajo:

- El modelo GRU demostró ser adecuado para la predicción de ventas en entornos industriales reales, logrando resultados satisfactorios en términos de variación interanual (YoY), que fue adoptada como métrica principal por su correspondencia con las prácticas actuales de planeación basada en juicio experto.
- Aunque en un inicio se planteó la modificación del periodo de pandemia debido a la presencia de valores atípicos, la experimentación mostró que dicha alteración no aporta mejoras significativas en el rendimiento del modelo. Por el contrario, mantener los datos originales permite preservar la integridad del conjunto histórico sin aumentar la complejidad del preprocesamiento.

- La segmentación temporal de los datos de entrenamiento resultó ser una estrategia efectiva. Se encontró que entrenar el modelo únicamente con información de los años más recientes (2021-2022) no solo reduce el tiempo de cómputo a la mitad, sino que produce resultados comparables e incluso superiores en algunos casos a los obtenidos al utilizar el histórico completo desde 2018.
- La integración de la tasa promedio de *scrap* en la predicción de producción representa un aporte práctico significativo, permitiendo generar un estimado más realista de las piezas a fabricar. No obstante, se observó un aumento en la variabilidad de las predicciones, lo cual sugiere la necesidad de seguir refinando esta incorporación mediante técnicas específicas para modelar el *scrap*.
- El uso del modelo GRU con un contexto reducido se mantuvo eficaz gracias a la naturaleza secuencial de los datos y al enfoque de predicción inmediata. Sin embargo, se reconoce que ampliar el contexto podría capturar dependencias temporales más complejas, siendo esta una línea de trabajo a considerar en futuras investigaciones.
- Finalmente, el modelo desarrollado cumple con los requerimientos del caso de estudio, automatizando una tarea crítica en la cadena de suministro industrial, disminuyendo la dependencia de métodos manuales, y ofreciendo una alternativa escalable, interpretable y operativamente viable para su futura implementación a mayor escala.

5.2. Recomendaciones

Con base en los resultados obtenidos y las conclusiones presentadas, se proponen las siguientes recomendaciones dirigidas a la aplicación práctica del modelo desarrollado, así como a futuras líneas de investigación que podrían complementar o extender esta propuesta:

- **Implementación progresiva en planta:** Se sugiere iniciar con una implementación piloto del modelo GRU para los productos de mayor volumen de ventas.

Esto permitirá validar su comportamiento en operación real, establecer criterios de confiabilidad y generar confianza en las áreas de planeación y producción antes de escalar a la totalidad del portafolio.

- **Automatización del flujo de datos:** Para asegurar la continuidad operativa del modelo, se recomienda establecer un flujo automatizado de actualización de datos, idealmente mediante integración directa con sistemas ERP (como SAP). Esto permitirá que el modelo trabaje con datos actualizados y reduzca la intervención manual.
- **Actualización periódica del modelo:** Dado que el comportamiento del mercado puede cambiar, se recomienda reentrenar el modelo de forma periódica (por ejemplo, cada trimestre o semestre), con los datos más recientes. Esto mantendrá la capacidad de adaptación del modelo frente a nuevas tendencias o cambios de comportamiento en la demanda.
- **Análisis separado del *scrap*:** Aunque se incorporó la tasa de *scrap* como una estimación estática, se recomienda desarrollar un modelo adicional que prediga dinámicamente el *scrap* por producto y línea. Esto permitiría una integración más precisa en la planeación de producción y facilitaría la detección de anomalías.
- **Exploración de contextos ampliados:** Si bien el contexto utilizado fue de tamaño 1, se sugiere explorar configuraciones con mayor profundidad temporal. Esto podría mejorar la capacidad del modelo para detectar patrones estacionales o multisemanales en productos con comportamiento periódico.
- **Evaluación de arquitecturas híbridas:** Para futuras investigaciones, se recomienda considerar modelos híbridos que combinen GRU con otros enfoques como LSTM, atención o redes convolucionales 1D. Estas combinaciones podrían capturar tanto la dinámica local como global de las series temporales.
- **Utilizar aprendizaje por refuerzo de retroalimentación humana (RLHF):** Este tipo de paradigma podría ajustarse al comportamiento del problema en cuestión, ya que, las fluctuaciones son imprevistas y el modelo tendría que tener

parámetros que se ajusten de acuerdo al experto humano, y que este experto evalúe las ventas atípicas.

Estas recomendaciones buscan facilitar la aplicación efectiva del modelo propuesto y sentar las bases para su evolución futura en un entorno real de manufactura, contribuyendo al desarrollo de sistemas de planeación más inteligentes, eficientes y orientados a datos.

Referencias

- Aggarwal, C. C. (2018). *Neural networks and deep learning : A textbook*. Springer.
- Alabi, O., y Bukola, T. (2023, 09). Introduction to descriptive statistics.. doi: 10.5772/intechopen.1002475
- AMIA. (2021). *Mexico en la produccion mundial de vehiculos*. Descargado de <https://www.amia.com.mx/about/vehiculos-mexico/>
- AMIA. (2023). *La produccion y exportacion de vehiculos ligeros de mexico mantienen su buen desempeno en febrero*. Descargado de <https://www.amia.com.mx/wp-content/uploads/2023/03/boletinPrensaFeb2023.pdf>
- Bandeira, S. G., Alcalá, S. G. S., Vita, R. O., y de Andrade Barbosa, T. M. G. (2020). Comparison of selection and combination strategies for demand forecasting methods. *Production*, 30. doi: 10.1590/0103-6513.20200009
- Berry, M. (1997). *Data mining techniques for marketing, sales, and customer support* (W. C. Publishing, Ed.).
- Bhat, N. P. (2023, 05). Optimizing scrap management in production planning processes. *International Journal of Computer Trends and Technology*, 71, 48-52. doi: 10.14445/22312803/IJCTT-V71I5P108
- Bishop, C. M. (2008). *Pattern recognition and machine learning (information science and statistics)*. Springer.
- Bode, C., Goetz, S., Schleich, B., y Wartzack, S. (2023, 06). Implementation and interpretation of a scrap and failure oriented multi-objective optimization considering operational wear. *Proceedings of the Design Society*, 3, 2225-2234. doi: 10.1017/pds.2023.223
- Bruce, P. (2020). *Practical statistics for data scientists* (N. Tache, Ed.). O'REILLY.
- Chen, Y., Xie, X., Pei, Z., Yi, W., Wang, C., Zhang, W., y Ji, Z. (2024, enero). Development of a time series e-commerce sales prediction method for short-shelf-life products using gru-lightgbm. *Applied Sciences*, 14(2), 866. doi: 10.3390/app14020866
- Dong, Y. (2023, 05). Descriptive statistics and its applications. *Highlights in Science, Engineering and Technology*, 47, 16-23. doi: 10.54097/hset.v47i.8159
- Ekman, M. (2021). *Learning deep learning*. Pearson Education, Limited.

- Enriquez, J. H. F. (2009). *Pronosticos de ventas para la administracion estrategica de los recursos en una empresa del sector automotriz*. Descargado de https://repositorio.tec.mx/bitstream/handle/11285/569469/DocsTec_10231.pdf?seq
- Fernando, M. L. (2021). *Sistema bi con prediccion de ventas basado en el algoritmo de series temporales para apoyar la gestion en la empresa motofuerza s.a.c.* (Tesis de Master no publicada). Universidad Catolica Santo Toribio de Mogrovejo.
- Flores, J. (2024a, junio). *Notebook: Salesbypn*. Descargado de <https://github.com/JManuelGithub/MasterDegreeCode/blob/main/Preprocessing/SalesbyPN.ipynb>
- Flores, J. (2024b, junio). *Sales forecasting*. Descargado de https://github.com/JManuelGithub/MasterDegreeCode/tree/main/Sales_forecasting
- Goodfellow, I., Bengio, Y., Courville, A., y Bach, F. (2017). *Deep learning*. MIT Press.
- Grus, J. (2019). *Data science from scratch first principles with python*. O'Reilly Media, Incorporated.
- Géron, A. (2019). *Hands-on machine learning with scikit-learn, keras, and tensorflow concepts, tools, and techniques to build intelligent systems*. O'Reilly Media.
- Hernandez, O. (2004). *Introduccion a la mineria de datos* (S. PEARSON EDUCACION, Ed.). Editorial Alhambra S. A. (SP).
- INEGI. (2016). *Estadisticas a proposito de la industria automotriz*. Descargado de https://www.inegi.org.mx/contenidos/productos/prod_serv/contenidos/espanol/bvinegi/productos/nueva_estruc/702825079963.pdf
- Keiron O'Shea, R. N. (2015). An introduction to convolutional neural networks. *CoRR*.
- Knafllic, C. N. (2015). *Storytelling with data*. Wiley.
- Krishnan, T., Khan, A., y Al-Qurni, J. (2022, 09). Aggregate production planning and scheduling in the industry 4.0 environment. *Procedia Computer Science*, 204, 784-793. doi: 10.1016/j.procs.2022.08.095
- Liu, B. (2006). *Web data mining*. Springer.
- Looney, C. G. (1997). *Pattern recognition using neural networks*. Oxford University Press.
- Lu, C., Shang, G., Xu, L., Shao, H., y Zhang, B. (2023, 08). Sales volume forecast of typical auto parts based on bigru: A case study. *E3S Web of Conferences*, 409. doi: 10.1051/e3sconf/202340904008

- Matonya, M., Kocsi, B., Pusztai, L., y Budai, I. (2020, 09). Production planning, scheduling and risk analysis in manufacturing operations by robotic process automation *. En *Proceedings of the 11th international conference on applied informatics*.
- Mogarala Guruvaya, A., Kollu, A., Divakarachari, P. B., Falkowski-Gilski, P., y Praveena, H. D. (2024, julio). Bi-gru-apso: Bi-directional gated recurrent unit with adaptive particle swarm optimization algorithm for sales forecasting in multi-channel retail. *Telecom*, 5(3), 537–555. doi: 10.3390/telecom5030028
- Olivares, K. G., Challú, C., Garza, F., Canseco, M. M., y Dubrawski, A. (2022). *NeuralForecast: User friendly state-of-the-art neural forecasting models*. PyCon Salt Lake City, Utah, US 2022. Descargado de <https://github.com/Nixtla/neuralforecast>
- Pack Kaelbling, L., y W. Moore, A. (1996). Reinforcement learning: A survey. *Journal of Artificial Intelligence Research* 4.
- Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. En *Advances in neural information processing systems* 32 (pp. 8024–8035). Curran Associates, Inc. Descargado de <http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf>
- Ramos, J. L. (2023). *Inflacion pegara a ventas automotrices, estima amda*. Descargado de <https://www.elsoldemexico.com.mx/finanzas/inflacion-pegara-a-ventas-automotrices-estima-amda-9901827.html#!>
- Raschka, S., Liu, Y. H., Mirjalili, V., y Dzhulgakov, D. (2022). *Machine learning with pytorch and scikit-learn*. Packt Publishing, Limited.
- Renna, P. (2022, 11). Special issue: the planning and scheduling of manufacturing systems. *Applied Sciences*, 12, 11713. doi: 10.3390/app122211713
- Riatma, D. L., Ari Roshinta, T., Masbahah, Rachman, Y. F., Azizul Hakimi, N., y Alike, R. (2025, enero). Comparison of deep learning algorithms (lstm and gru) for predicting product stocks. En *2025 international conference on computer sciences, engineering, and technology innovation (icocseti)* (pp. 1–6). IEEE. doi: 10.1109/icocseti63724.2025.11019594
- Rottoli, G. (2018). Formulacion de procesos para una ingenieria de explotacion de informacion espacial. En *Formulacion de procesos para una ingenieria de*

explotacion de informacion espacial.

SAP. (2024). *What is sap?* Descargado de <https://www.sap.com/about/what-is-sap.html>

SE. (2010). *Immex*. Descargado de <http://www.2006-2012.economia.gob.mx/comunidad-negocios/industria-y-comercio/instrumentos-de-comercio-exterior/immex>

Smirnov, P., y Sudakov, V. (2021, 05). Forecasting new product demand using machine learning. *Journal of Physics: Conference Series*, 1925, 012033. doi: 10.1088/1742-6596/1925/1/012033

VanderPlas, J. (2017). *Python data science handbook* (D. Schanafelt, Ed.). O'REILLY.

Veracruzana, U. (2023). *Toma de tiempo ciclo*. Descargado de <https://www.uv.mx/iiesca/files/2020/09/01CA2020-01.pdf>

Zarco, J. L. (2021). *La pandemia del covid19 y la industria automotriz en mexico. un balance parcial de danos colaterales*. Descargado de <http://ru.iiec.unam.mx/5484/1/110-Linares.pdf>