

Universidad Nacional del Este.
Facultad Politécnica.



Sistema de compra inteligente basado en el histórico de ventas.

Valeria Soledad Arevalos Arevalos
y Marcelo Andrés González Arias

Año 2023.

Universidad Nacional del Este
Facultad Politécnica

Carrera Ingeniería de Sistemas.
Cátedra Trabajo Final de Grado II.

Sistema de compra inteligente basado en el histórico de ventas.

Por: **Valeria Soledad Arevalos Arevalos**
y **Marcelo Andrés González Arias**

Profesor Orientador: **Lic. Roberto Alfredo Demestri Rigoni.**

Trabajo final de grado presentado a la Facultad Politécnica de la Universidad Nacional del Este como parte de los requisitos para optar al título de Ingeniero de Sistemas.

Ciudad del Este, Alto Paraná. Paraguay.

Julio 2023

FICHA CATALOGÁFICA
BIBLIOTECA DE LA FACULTAD POLITÉCNICA
DE LA UNIVERSIDAD NACIONAL DEL ESTE

Arevalos Arevalos, Valeria Soledad 1998.
Sistema de compra inteligente basado en el histórico de ventas
Ciudad del Este, Alto Paraná. Año: 2023.
Páginas: 81 .

González Arias, Marcelo Andrés 1996
Sistema de compra inteligente basado en el histórico de ventas
Ciudad del Este, Alto Paraná. Año: 2023.
Páginas: 81 .

Orientador: Lic. Roberto Alfredo Demestri Rigoni.
Área de estudio: Tecnológica.
Carrera: Ingeniería de Sistemas.
Titulación: Ingeniero de Sistemas.
Trabajo Final de Grado. Universidad Nacional del Este,
Facultad Politécnica.

Descriptores: 1. Valeria Arevalos. 2. Marcelo González
Smart purchasing system based on sales history.
Key words: 1. Input Demand Forecast, 2. Matematical Modelling
3. Artificial Neural network.

Yo, Lic. Roberto Alfredo Demestri Rigoni, documento de identidad No. 1.161.154, Profesor Orientador del TFG titulado “*Sistema de compra inteligente basado en el histórico de ventas*”, del Alumno Marcelo Andrés González Arias, documento de identidad No. 5.449.594 y de la Alumna Valeria Soledad Arevalos Arevalos, documento de identidad No. 5.040.149 de la carrera Ingeniería de Sistemas de la Facultad Politécnica de la Universidad Nacional del Este; certifico que el mencionado Trabajo Final de Grado ha sido realizado por dicho Alumno, de lo cual doy fe y en mi opinión reúne las condiciones para su presentación y defensa ante la Mesa Examinadora designada por la institución.

Ciudad del Este, ____ de _____ de ____

Lic. Roberto Alfredo Demestri Rigoni

Nosotros, los miembros de la Mesa Examinadora del Trabajo Final de Grado titulado “*Sistema de compra inteligente basado en el histórico de ventas*”, de la carrera Ingeniería de Sistemas de la Facultad Politécnica de la Universidad Nacional del Este, hacemos constar que el citado trabajo ha sido evaluado en fondo y forma por esta Mesa, la que por _____ ha resuelto asignar la calificación _____

Ciudad del Este, ____ de _____ de ____

Profesor _____
Presidente de la Mesa Examinadora

Profesor _____
Miembro de la Mesa Examinadora

Profesor _____
Miembro de la Mesa Examinadora

Dedicado a Dios, mi fuente de fortaleza y guía en cada paso de mi vida. A mis queridos padres, cuyo amor incondicional y apoyo constante han sido mi mayor motivación para seguir adelante. A mi esposo, por ser mi compañero inquebrantable y por su apoyo constante en esta travesía. Este logro es un reflejo del amor y la dedicación de mi familia. A todos ellos, les expreso mi profundo agradecimiento.

Con profunda devoción, dedico este trabajo a la guía divina de Dios, rindiendo homenaje a la memoria amorosa y siempre viva de mi madre. Agradezco de todo corazón el incondicional respaldo de mi padre, cuya fortaleza guió cada paso de este proceso. Mis hermanos, fieles compañeros en cada instante, son la esencia misma de este logro; sin su constante presencia, esta meta habría sido inalcanzable.

Deseamos expresar nuestro sincero agradecimiento al Lic. Roberto Alfredo Demestri Rigoni, mi tutor, por su apoyo constante y paciencia incansable a lo largo de este proyecto. Su experiencia y dedicación fueron esenciales para llevar este trabajo a su exitosa conclusión. También quiero extender mi agradecimiento a todos los profesores que han sido parte de mi recorrido académico.

*Aprender es el principio de la vida, pero la humildad es el
comienzo de la sabiduría. - San Vicente de Paul*

Resumen

Este trabajo propone el desarrollo de un sistema inteligente de compras para restaurantes, que se basa en un modelo matemático de predicción de la demanda de insumos. Este modelo utiliza técnicas estadísticas avanzadas y métodos de pronóstico para analizar patrones de consumo pasados y prever con precisión las cantidades de insumos necesarias en el futuro. Además, se incorpora una red neuronal entrenada para mejorar aún más las estimaciones de demanda, permitiendo así reducir el desperdicio de alimentos, optimizar los niveles de inventario y mejorar la eficiencia operativa del restaurante. La metodología de investigación involucra la recopilación y análisis de datos históricos de ventas y compras, así como la exploración de diversas técnicas de pronóstico y algoritmos de redes neuronales. Se llevarán a cabo experimentos para evaluar la eficacia del sistema en términos de reducción de costos y gestión de inventario.

En última instancia, esta tesis busca contribuir al desarrollo de un sistema de compras efectivo que permita a los restaurantes tomar decisiones informadas y oportunas en cuanto a la adquisición de insumos. La combinación de un modelo matemático de predicción de demanda con redes neuronales representa un enfoque innovador para abordar los desafíos de la gestión de inventario en la industria gastronómica, con el objetivo de mejorar la eficiencia y sostenibilidad operativa.

Descriptores: 1. Predicción de la demanda de insumos, 2. Modelo matemático, 3. Redes neuronales artificiales.

Abstract

This work proposes the development of an intelligent purchasing system for restaurants, based on a mathematical demand prediction model. This model utilizes advanced statistical techniques and forecasting methods to analyze past consumption patterns and accurately forecast the quantities of inputs needed in the future. Additionally, a trained neural network is incorporated to further enhance demand estimations, thereby allowing for the reduction of food waste, optimization of inventory levels, and improvement in restaurant operational efficiency. The research methodology involves the collection and analysis of historical sales and purchasing data, as well as the exploration of various forecasting techniques and neural network algorithms. Experiments will be conducted to assess the system's effectiveness in terms of cost reduction, inventory management, and overall customer satisfaction.

Ultimately, this thesis aims to contribute to the development of an effective purchasing system that enables restaurants to make informed and timely decisions regarding input procurement. The combination of a mathematical demand prediction model with neural networks represents an innovative approach to addressing inventory management challenges in the gastronomy industry, with the goal of enhancing operational efficiency and sustainability.

Key words: 1. Input Demand Forecast , 2. Matematical Modelling, 3. Artificial Neural network.

Índice general

Resumen	x
Abstract	xi
Índice de figuras	xv
Índice de tablas	xvi
Acrónimos y símbolos	xvi
1. Introducción	1
1.1. Motivación	1
1.2. Definición del problema	2
1.3. Objetivos	2
1.3.1. General	2
1.3.2. Específico	2
1.4. Hipótesis	2
1.5. Fundamentación	3
1.6. Impacto de la investigación	3
1.7. Organización del trabajo.	3
2. Conceptos fundamentales, teorías y antecedentes	5
2.1. Predicciones de Compras	5
2.1.1. Relación de Demanda y Gestión de Suministro	5
2.1.2. Importancia de la Predicción de Compra de Insumos	5
2.1.3. Factores que Influyen en la Curva de Insumos Deman- dados	8
2.2. Tiempos Actuales en las Predicciones de Demanda	8
2.3. Sistema de información	9
2.4. Minería de datos	12
2.5. Técnicas de modelado de procesos de ETL	13
2.6. Inteligencia Artificial	14

2.6.1.	El aprendizaje automático (<i>machine learning</i>)	14
2.6.2.	Aprendizaje supervisado	16
2.6.3.	Aprendizaje no supervisado	17
2.6.4.	Aprendizaje de refuerzo	17
2.6.5.	Redes Neuronales Artificiales	18
2.6.6.	Capas de la neurona artificial	20
2.6.7.	Modelo de redes neuronales artificiales	21
2.6.8.	Arquitectura de una red neuronal	21
2.6.9.	Perceptron	22
2.6.10.	Perceptrón multica	23
2.6.11.	Redes neuronales recurrentes (RNN)	23
2.6.12.	Arquitectura básica de una RNN	24
2.6.13.	Redes de Memoria Corta y Larga LSTM	25
2.6.14.	Modelo de Regresión Lineal	26
2.6.15.	Los pesos sinápticos	27
2.7.	Métricas para la evaluación de una red neuronal	28
2.7.1.	El error medio cuadrático (MSE)	28
2.7.2.	Raíz del error medio cuadrático (RMSE)	28
2.7.3.	Error Absoluto Medio (MAE)	29
2.7.4.	Error porcentual absoluto medio (MAPE)	30
2.7.5.	El coeficiente de determinación R^2	30
2.8.	Antecedentes	31
3.	Método, enfoque e implementación	33
3.1.	Método	33
3.2.	Preprocesamiento de datos	33
3.2.1.	Obtención de datos	33
3.2.2.	Modelado de datos	35
3.2.3.	Escalado de datos	37
3.2.4.	Análisis de los datos	38
3.2.5.	Definición de las variables de entrada y salida	39
3.2.6.	Lectura y escalado del dataset en Python	41
3.2.7.	Distribución del conjunto de datos	44
3.3.	Modelos de predicción	45
3.3.1.	Modelo de Regresión lineal	46
3.3.2.	Modelo LSTM	49
3.4.	Prototipo del sistema	55

4. Resultados	58
4.1. Métricas de rendimiento	58
4.1.1. Modelo de regresión lineal	58
4.1.2. Modelo LSTM	59
4.2. Resultados de predicción de compras	62
5. Discusión	64
5.1. Logros alcanzados	64
5.2. Solución del problema de investigación	65
5.2.1. General	65
5.2.2. Específico	65
5.3. Análisis de hipótesis	66
5.4. Sugerencias para futuras investigaciones	67
Anexos	68
Referencias bibliográficas	77

Índice de figuras

2.1. Predicción de demanda a corto, mediano y largo plazo.	6
2.2. Proceso de Optimización de inventario.	7
2.3. Arquitectura de un sistema experto.	12
2.4. flujo de trabajo típico para el uso del aprendizaje automático en modelado predictivo	16
2.5. Aprendizaje supervisado.	17
2.6. Componentes de una neurona.	19
2.7. Tipos de neuronas artificiales.	20
2.8. Aplicación de la Red Perceptron	23
2.9. La arquitectura básica de una RNN	25
2.10. Representación de una celda LSTM.	26
2.11. Regresión lineal	27
3.1. Estructura inicial de la tabla productos.csv	34
3.2. Estructura inicial de la tabla ventas.csv	34
3.3. Datos de la tabla productos(Los primeros 5 productos)	36
3.4. Datos de la tabla ventas	37
3.5. Datos escalados(muestra de la tabla ventas)	38
3.6. Serie temporal de las ventas totales por día	39
3.7. Tabla resultante con las variables de entrada	41
3.8. Código de lectura y escalado del dataset en Python	43
3.9. Gráfico en violín incluyendo los tres set train, validación y test	44
3.10. Gráfico en violín de los datos de salida escaladas	44
3.11. Código de distribución de los datos.	45
3.12. Distribución de los datos, ventas totales por día.	45
3.13. Código de la estructura de regresión lineal.	48
3.14. Gráfico de líneas con Matplotlib.	49
3.15. Código de la estructura del LSTM.	50
3.16. Código de la estructura del LSTM datos de prueba 1.	51
3.17. Rendimiento de datos de prueba 1.	52
3.18. Gráfico de predicción LSTM prueba 1	53

3.19. Codigo de la estructura del LSTM datos de prueba 2.	54
3.20. Rendimiento de datos de prueba 2.	55
3.21. Grafico de predicción LSTM prueba 2	55
4.1. Algoritmo de la estructura de regresión lineal.	59
4.2. Metrica de error regresión lineal.	59
4.3. Algoritmo de estructura LSTM.	60
4.4. Grafico de predicción con LSTM.	61
4.5. Comparativo de desempeños LSTM.	61
4.6. Grafico de perdida en las épocas LSTM.	62
5.1. Diagrama de clases sistema de gestion de inventario.	68
5.2. Login del sistema	69
5.3. Interfaz productos	70
5.4. Interfaz de carga de ingredientes	70
5.5. Codigo carga de productos e ingresientes	71
5.6. Interfaz de compras	72
5.7. Codigo carga de compras	72
5.8. Interfaz de ventas	73
5.9. Codigo carga de ventas	73
5.10. Interfaz de lista de predicciones hechas	74
5.11. Informe de predicción de ingredientes a comprar	74
5.12. Expotación de Python	75
5.13. Importación a javascript	75
5.14. Importación a javascript	76

Índice de Tablas

3.1. Descripción de datos de la tabla de productos	36
3.2. Descripción de datos de la tabla ventas	37
3.3. Descripción de datos escalados de la tabla ventas	38
3.4. Variables de entrada	40
4.1. Tabla de resultados de las predicciones de ambos modelos . . .	62

Capítulo 1

Introducción

En la industria gastronómica, la gestión eficiente de insumos es clave para el éxito de un restaurante. Para lograrlo, es esencial prever y ajustar la demanda de insumos de manera precisa. Si bien es común que los restaurantes busquen mejorar sus sistemas de gestión utilizando principios de empresas industriales, es importante recordar que la industria de la restauración tiene particularidades únicas, como la atención al cliente, el control de costos y la calidad de los alimentos y bebidas, que deben ser tenidas en cuenta para una gestión efectiva.

Un subsistema vital en un restaurante es el de compras, que se encarga de asegurar la calidad de los proveedores y el suministro constante de productos de alta calidad. Aquí es donde la tecnología computacional desempeña un papel crucial. Los sistemas avanzados permiten procesar grandes cantidades de datos y realizar predicciones precisas, lo que contribuye significativamente a optimizar la gestión de insumos y, por lo tanto, al éxito operativo y financiero del restaurante.

1.1. Motivación

Mediante la aplicación de técnicas avanzadas de análisis de datos y aprendizaje automático, este sistema proporcionará recomendaciones precisas y estratégicas, contribuyendo así al avance de la inteligencia empresarial y ofreciendo una herramienta eficiente para la toma de decisiones informadas y fundamentadas. El objetivo final es mejorar la eficiencia operativa, reducir costos y alcanzar ventajas competitivas, permitiendo a las empresas adaptarse y prosperar en un entorno comercial dinámico y desafiante.

1.2. Definición del problema

Este sistema debe ser capaz de utilizar el histórico de ventas del local para predecir la demanda futura de los productos, y en consecuencia, realizar las compras de manera más eficiente y rentable para la gerencia del local.

Problema de investigación: *Necesidad de optimizar la gestión de compras de materia prima de un local gastronómico en Ciudad del Este, utilizando sistema de compras inteligente basado en el histórico de ventas.*

1.3. Objetivos

1.3.1. General

Desarrollar un sistema de compra inteligente basado en histórico de ventas para optimizar la gestión de compras de una empresa gastronómica de Ciudad del Este mediante algoritmos y técnicas de análisis de datos para predecir la demanda de productos en función del comportamiento de los clientes.

1.3.2. Específico

1. Comprender en profundidad la lógica de negocio de la empresa gastronómica para identificar los procesos críticos que deben ser incluidos en el sistema informático a desarrollar.
2. Recabar los requisitos del sistema con los usuarios y partes interesadas.
3. Modelar la lógica del sistema informático, utilizando técnicas y herramientas adecuadas, para garantizar su integridad y eficiencia.
4. Codificar el modelo lógico definido en lenguajes de programación Python y PHP.
5. Depurar los datos cuantitativos recopilados sobre el comportamiento del consumidor.
6. Realizar pruebas de usabilidad, accesibilidad, multiplataforma, y escalabilidad.

1.4. Hipótesis

La implementación de un sistema inteligente en un local gastronómico de Ciudad del Este, permitirá una gestión de compras cuya eficiencia logrará

reducir el costo de materia prima en al menos 10 % y predecir al menos con un 70 % de acierto, la variedad y cantidad de productos a ser comprados.

1.5. Fundamentación

En el entorno comercial actual, la toma de decisiones basada en información histórica puede ser un factor clave para el éxito de una empresa. Sin embargo, en muchos casos, esta información no se explota adecuadamente y la gestión de datos se convierte en una tarea compleja.

En el caso de una empresa gastronómica, es importante conocer el rendimiento en ventas y controlar los índices que reflejen el desempeño del negocio. Sin embargo, la simple gestión de estos índices no es suficiente, ya que la empresa puede planificar una determinada cantidad de ventas y adquirir una cantidad de materia prima en consecuencia, pero luego no lograr vender todo lo adquirido y mantener un exceso de stock, o incluso tener productos en inventario que no se rotan.

Es por eso que se plantea el desarrollo de un sistema que permita proyectar las ventas, para tener un presupuesto real de compras que se adapte a la demanda del mercado y evite el exceso de stock o la falta de rotación de inventarios. De esta manera, la empresa podrá tomar decisiones informadas y maximizar su rendimiento.

1.6. Impacto de la investigación

La implementación del sistema inteligente de compra basado en el histórico de ventas permitirá al gerente de la empresa tomar decisiones informadas y estratégicas, lo que se traducirá en un mejor desempeño económico y financiero de la empresa gastronómica.

Además, la investigación realizada para el desarrollo del sistema también tendrá un impacto positivo en otros estudiantes y futuros proyectos, ya que se podrán utilizar los conocimientos y experiencias adquiridos para mejorar otros procesos y aplicaciones en el ámbito empresarial y tecnológico. En consecuencia, la investigación tendrá un impacto significativo tanto en la empresa como en el ámbito académico y profesional.

1.7. Organización del trabajo.

El presente trabajo está organizado de la siguiente manera:

Capítulo 1: Este capítulo introductorio presenta la motivación para la investigación, que se enfoca en la gestión eficiente de insumos en la industria gastronómica. El problema de investigación se centra en la optimización de la gestión de compras de materia prima en un restaurante de Ciudad del Este. Se establecen los objetivos, hipótesis y la importancia del estudio, destacando su impacto positivo en la empresa y en futuros proyectos.

Capítulo 2: Se presentan algunos conceptos relacionados con las predicciones de compras de insumos, cómo funciona la relación de demanda y gestión de suministro de un local gastronómico, por qué es importante la predicción de compra de insumos para el planeamiento futuro tanto a corto, mediano y largo plazo, los factores que influyen en el comportamiento de la curva de insumos demandados y tiempos actuales en las predicciones de demanda.

Capítulo 3: En este capítulo se aborda la afinación de hiperparámetros en un modelo de predicción, utilizando el porcentaje de pérdida Root Mean Square (RMS) como métrica guía para evaluar el rendimiento del modelo. Se describen los procedimientos para ajustar diferentes hiperparámetros, como la tasa de aprendizaje y el número de capas ocultas, con el objetivo de minimizar la pérdida RMS y, por lo tanto, mejorar la precisión y eficiencia del modelo de predicción. Se detalla la metodología empleada en este proceso, incluyendo la utilización de conjuntos de validación y estrategias de búsqueda de hiperparámetros, y se presentan los resultados obtenidos tras la afinación.

Capítulo 4: En este capítulo se presentan y analizan los resultados de los hiperparámetros y de las predicciones de compras con los modelos de predicción estudiados.

Capítulo 5: En el último capítulo de esta tesis, se procede a la discusión y análisis de los resultados obtenidos. Se lleva a cabo una evaluación de la eficacia del modelo de predicción y se reflexiona sobre su relevancia en el marco del sistema de compra inteligente basado en el historial de ventas. Además, se destacan áreas potenciales para investigaciones futuras en el contexto de este sistema.

Capítulo 2

Conceptos fundamentales, teorías y antecedentes

2.1. Predicciones de Compras

La predicción es el proceso de anticipar la cantidad y tipo de insumos que un negocio gastronómico necesitará adquirir en el futuro, con el objetivo de asegurar un suministro adecuado y eficiente.

2.1.1. Relación de Demanda y Gestión de Suministro

La relación de demanda y gestión de suministro se refiere a la interacción entre la cantidad de productos o insumos que los clientes requieren (demanda) y cómo la empresa se asegura de tener suficientes suministros disponibles para satisfacer esa demanda de manera eficiente. En un negocio gastronómico, la gestión de suministros es crucial para evitar situaciones en las que falten ingredientes clave, lo que podría afectar negativamente la calidad del servicio y la satisfacción del cliente.

2.1.2. Importancia de la Predicción de Compra de Insumos

La mayoría de las empresas distribuidoras de productos sufren de manera recurrente al no conocer la cantidad o un aproximado de productos que debería mantener en stock ya que, por un lado, si el stock es demasiado grande se pueden producir pérdidas de mercancía y costos innecesarios de transporte y almacenamiento, por el contrario, si el stock es demasiado pequeño, este será insuficiente para cubrir la demanda de los diferentes productos y se verá

reflejado en la pérdida de clientes [1].

La aplicación de un modelo de predicción para planificar la demanda futura es un proceso circular de mejora continua. Eso significa que el modelo es enriquecido constantemente con datos en tiempo real para realizar predicciones más precisas y generar una planificación más acorde a la realidad [2].



Figura 2.1: Predicción de demanda a corto, mediano y largo plazo.

1. Utiliza la **Predicción de la demanda a largo plazo para la planificación estratégica**. Simula y compara diferentes escenarios hipotéticos de demanda, anticipa los cambios de mercado y planifica la contratación futura de recursos.
2. Utiliza la **Predicción de la demanda a corto plazo para la planificación operativa**. Planifica semanalmente las operaciones y recursos del día.
3. Utiliza la **Predicción de la demanda a medio plazo para la planificación táctica**. Conoce las capacidades con meses de antelación y compáralas con las actuales para detectar posibles necesidades.

La predicción de compra de insumos es vital para la planificación a corto, mediano y largo plazo en un negocio gastronómico por varias razones:

1. **Optimización de Inventario:** El inventario tiene como propósito fundamental proveer a la empresa de materiales necesarios, para su continuo y regular desenvolvimiento, es decir, el inventario tiene un papel

vital para el funcionamiento acorde y coherente dentro del proceso de producción y de esta forma afrontar la demanda [3].

Al prever la demanda futura, el negocio puede mantener un nivel de inventario óptimo. Comprar en exceso puede llevar al desperdicio de alimentos, mientras que comprar muy poco puede resultar en escasez y pérdida de ventas.



Figura 2.2: Proceso de Optimización de inventario.

2. **Reducción de Costos:** Una predicción precisa permite comprar solo lo necesario, lo que reduce los costos asociados con el almacenamiento y la conservación de productos perecederos.

Si se mantienen inventarios demasiado altos, el costo podría llevar a una empresa a tener problemas de liquidez financiera, esto ocurre porque un inventario “parado” inmoviliza recursos que podrían ser mejor utilizados en funciones más productivas de la organización [3].

3. **Eficiencia Operativa:** Saber qué insumos se necesitarán en el futuro permite planificar y programar las operaciones de manera más eficiente, evitando retrasos y problemas logísticos.

Es útil mantener los inventarios en las empresas porque, se tiene en cuenta la capacidad de predicción con el fin de planear la capacidad y establecer un cronograma de producción, también fluctuaciones en la

demanda ósea una reserva de inventarios a la mano que supone protección, inestabilidad de los suministros, protección deprecios, descuentos por cantidad, menores costos de pedidos [3].

4. **Satisfacción del Cliente:** Mantener un suministro constante de productos esencial para el negocio, ya que los clientes esperan encontrar su elección preferida en el menú en todo momento.

2.1.3. Factores que Influyen en la Curva de Insumos Demandados

Varios factores pueden influir en el comportamiento de la curva de insumos demandados:

- **Día de la Semana y Estacionalidad:** Los patrones de consumo pueden variar según el día de la semana y la temporada. Por ejemplo, los fines de semana pueden tener una mayor demanda en comparación con los días laborables.
- **Eventos Especiales:** Eventos como días festivos, celebraciones locales o conciertos cercanos pueden aumentar la demanda de alimentos y bebidas.
- **Tendencias de Consumo:** Las tendencias gastronómicas y las preferencias cambiantes de los consumidores pueden afectar la demanda de ciertos productos.
- **Clima:** Las condiciones climáticas también pueden influir en la demanda. Por ejemplo, un día caluroso podría aumentar la demanda de bebidas frías.
- **Promociones y Ofertas:** Las promociones especiales pueden aumentar temporalmente la demanda de ciertos productos.

2.2. Tiempos Actuales en las Predicciones de Demanda

En la actualidad, las predicciones de demanda se benefician ampliamente de las tecnologías avanzadas y el análisis de datos. Las empresas pueden aprovechar sistemas de gestión de inventario y software de análisis de datos

para recopilar y procesar información histórica y en tiempo real. Estas herramientas les permiten aplicar técnicas estadísticas y modelos de pronóstico para anticipar con mayor precisión los patrones de demanda futura.

En tiempos recientes, se ha observado la aplicación de diversas técnicas en el campo de la inteligencia artificial, como sistemas expertos, y más recientemente, algoritmos genéticos. Sin embargo, a pesar de esta evolución, los modelos que han captado una atención destacada son los basados en Redes Neuronales Artificiales (RNAs). Con el transcurso del tiempo, se han desarrollado múltiples arquitecturas de RNAs para abordar una variedad de problemas en el ámbito de la predicción de demanda.

2.3. Sistema de información

Según [4] el sistema de información *“Es un conjunto de elementos que interactúan entre sí, con el fin de apoyar las actividades de una empresa o negocio”*.

Es importante tener en cuenta que la necesidad de información en las organizaciones es vital para alcanzar el éxito y que un sistema de información debe justificar su implementación desde el punto de vista costo/beneficio, basándose en el valor que se le otorga a la información dentro de la organización [4]. Los beneficios pueden ser tangibles e intangibles, y dependen de los objetivos y necesidades de la organización.

Los sistemas de información se desarrollan para diferentes propósitos, según las necesidades de los usuarios humanos y la empresa. En definitiva, el uso adecuado de la información y la implementación de sistemas de información efectivos pueden marcar una gran diferencia en el éxito de una organización.

Tipos de sistema de información

El propósito de un sistema de información, puede ser muy amplio, todo depende de las necesidades de la organización. Existen distintos tipos de sistemas de información, entre los que destacan los siguientes [4]:

- **Sistemas de procesamiento de transacciones**

Se define como transacción un suceso que implica o afecta a una organización, y que está compuesta por datos referentes a ellas y que son de importancia para la organización [4]. Estos sistemas se encargan del procesamiento de los datos referentes a las transacciones, además de permitir la automatización de tareas y procesos operativos.

La información que se obtiene como salida es utilizada posteriormente por los funcionarios de nivel operativo de la organización en la toma de decisiones.

Las razones para el procesamiento de las transacciones son:

- Clasificación: Implica agrupar todos los datos de acuerdo con características comunes.
- Operaciones de cálculo: Consiste en realizar alguna operación para obtener resultados útiles.
- Ordenamiento: Consiste en disponerlos de alguna forma o secuencia, facilita el procesamiento y la búsqueda.
- Síntesis: Reduce los datos en información breve y concisa.
- Almacenamiento: Permite el registro de todas y cada una del suceso que afectan a la organización.

■ Sistema de información administrativa

Los sistemas de información administrativa (MIS) no sustituyen a los sistemas de procesamiento de transacciones; más bien, todos los sistemas MIS incluyen el procesamiento de transacciones [4]. Los MIS son sistemas de información computarizados que funcionan debido a la decidida interacción entre las personas y las computadoras.

Al requerir que las personas, el software y el hardware funcionen en concierto, los sistemas de información administrativa brindan soporte a los usuarios para realizar un espectro más amplio de tareas organizacionales que los sistemas de procesamiento de transacciones, incluyendo los procesos de análisis y toma de decisiones. Para acceder a la información, los usuarios del sistema de información administrativa comparten una base de datos común; esta almacena tanto los datos como los modelos que permiten al usuario interactuar con ellos, interpretarlos y aplicarlos. Los sistemas de información administrativa producen información que se utiliza en el proceso de toma de decisiones. También pueden ayudar a integrar algunas de las funciones de información computarizadas de una empresa.

■ Sistema de soporte de decisiones DSS

Son sistemas de información que tienen como propósito auxiliar al usuario con las decisiones únicas que no se repiten y que no tienen una estructura definida [4]. Además de estar hechos a la medida de la persona

o grupo que los usa en comparación con los Sistemas de información Gerencial. El propósito de estos sistemas es el de responder correctamente a condiciones inesperadas y propias de la información. Esto permite que sean empleados en niveles altos de la organización.

■ **Sistema de información gerencial**

Los Sistemas de Información Gerencial, también llamados Sistemas de Reportes de Gerencia, se dedican al apoyo de decisiones siempre que los requerimientos de información sean identificados, esto es, que la información que necesita para la toma de decisiones haya sido analizada anteriormente, y que esta misma decisión pueda tomarse nuevamente [4]. Estos sistemas pueden extraer la información necesaria de cualquier parte de la organización, por lo que la información necesaria ya se tiene almacenada al ser procesada por un sistema de transacciones.

■ **Sistema experto e inteligencia artificial**

La inteligencia artificial (IA) puede ser considerada como el campo dominante de los sistemas expertos. La idea general de la IA ha sido desarrollar equipos que se comporten de manera inteligente [4].

Dos ramas de investigación de la IA son:

- La comprensión del lenguaje natural.
- El análisis de la habilidad para razonar un problema y llegar a una conclusión lógica.

Los sistemas expertos utilizan las metodologías de razonamiento de la IA para resolver los problemas que los usuarios de negocios (y otros tipos de usuarios) les presentan. Los sistemas expertos son una clase muy especial de sistema de información que ha demostrado su utilidad comercial gracias a la disponibilidad extendida de hardware y software como las computadoras personales (PC) y las interfaces de sistemas expertos.

Un sistema experto (también conocido como sistema basado en el conocimiento) captura y utiliza en forma efectiva el conocimiento de uno o varios expertos humanos para resolver un problema específico al que una organización se enfrenta. Cabe mencionar que a diferencia de los sistemas DSS, que en última instancia dejan la decisión a la persona encargada de la toma de decisiones, un sistema experto selecciona la mejor solución para un problema o una clase específica de problemas.

Los componentes básicos de un sistema experto son la base de conocimiento, un motor de inferencia que conecta al usuario con el sistema mediante el proceso de consultas en lenguajes, como el lenguaje de consulta estructurado (SQL), y la interfaz de usuario. Las personas conocidas como ingenieros del conocimiento capturan la experiencia de los expertos, crean un sistema computacional que incluye este conocimiento y después lo implementan [4].

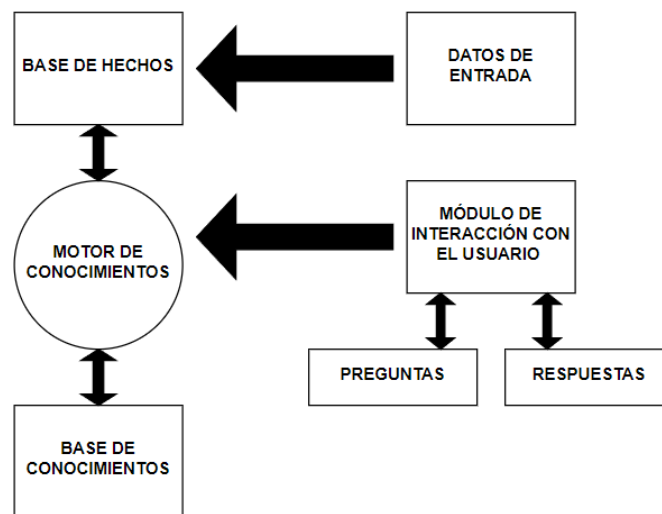


Figura 2.3: Arquitectura de un sistema experto.

En los Sistemas Expertos el conocimiento se hace explícito en forma de reglas, en la computación neuronal las ANN generan sus propias reglas aprendiendo de los ejemplos que se les muestran en la fase de entrenamiento [5].

2.4. Minería de datos

La minería de datos puede definirse inicialmente como un proceso de descubrimiento de nuevas y significativas relaciones, patrones y tendencias al examinar grandes cantidades de datos [6].

El análisis de datos, potenciado por herramientas informáticas, ha dado lugar a la minería de datos. Esta disciplina busca descubrir automáticamente

conocimiento en grandes bases de datos, identificando patrones, tendencias y perfiles a través de tecnologías avanzadas como el reconocimiento de patrones, redes neuronales y algoritmos genéticos.

Inicialmente, los sistemas de información se centraban en recopilar datos para apoyar la toma de decisiones. Con la informatización de las organizaciones, se amplió su función para respaldar los procesos esenciales. Ahora, se buscan prestaciones adicionales, como sistemas de información para la toma de decisiones.

La Minería de Datos se puede ubicar en el nivel más alto de la evolución de los procesos tecnológicos de análisis de datos [7].

La Minería de Datos involucra e integra técnicas de diferentes disciplinas tales como tecnologías de bases de datos y data warehouse, estadística, aprendizaje de máquina, computación de alta performance, computación evolutiva, reconocimiento de patrones, redes neuronales, visualización de datos, recuperación de información, procesamiento de imágenes y señales, y análisis de datos espaciales o temporales [8].

2.5. Técnicas de modelado de procesos de ETL

Las herramientas de Extracción-Transformación-Carga (ETL) son piezas de software responsables de la extracción de datos de varias fuentes, su limpieza, personalización e inserción en un almacén de datos [9].

El proceso de extracción, transformación y carga – ETL es una de las actividades técnicas más críticas en el desarrollo de soluciones de inteligencia de negocios BI [10].

El proceso de ETL es esencial para garantizar la calidad y la integridad de los datos antes de que se utilicen en aplicaciones analíticas o de toma de decisiones. Permite a las organizaciones transformar datos crudos en información procesable y confiable. Además, con la creciente cantidad de datos disponibles en la actualidad, el ETL se ha vuelto cada vez más importante para integrar datos de diversas fuentes y garantizar que estén listos para su análisis.

Para autores como Ralph Kimball los Almacenes de Datos son “una copia de los datos transaccionales estructurados específicamente para consultas y

análisis”, mientras que Bill Inmon define el término Almacén de Datos como: “una colección de datos orientados por temas, integrados, variables en el tiempo y no volátiles para el apoyo de la toma de decisiones”. Los Almacenes de Datos son integradores, ya que su contenido proviene de diversas fuentes de datos como: Sistemas heredados, Archivos de Textos, Base de Datos Relacionales, ERP, entre otras posibilidades. La forma de lograr esta integración es a través del uso y desarrollo de los Procesos ETL. Estos procesos son los encargados de la extracción de los datos desde sus fuentes de origen, de transformarlos a la información deseada, de lograr la limpieza necesaria en aquellos que lo requieran y finalmente cargar al Almacén de Datos deseado, el que será utilizado con alguna finalidad como análisis en un área de las ventas de una corporación o el estudio de tendencias de alguna consultora [11].

2.6. Inteligencia Artificial

2.6.1. El aprendizaje automático (*machine learning*)

El aprendizaje automático como area de las ciencias computacionales aplicadas, desarrolla algoritmos capaces de tomar datos numéricos y alfanuméricos almacenados en un computador [12].

Según Arthur Samuel, el aprendizaje automático se define como el campo de estudio que otorga a las computadoras la capacidad de aprender sin ser programadas explícitamente [13]. Es decir en lugar de seguir instrucciones detalladas, las computadoras pueden mejorar su desempeño mediante la adquisición de conocimiento y habilidades a partir de datos y experiencias previas

El aprendizaje automático se utiliza para enseñar a las máquinas a manejar los datos de manera más eficiente. A veces, después de ver los datos, no podemos interpretar la información extraída de los mismos. En ese caso, aplicamos el aprendizaje automático.

Con la abundancia de conjuntos de datos disponibles, la demanda de aprendizaje automático está en aumento. Muchas industrias aplican el aprendizaje automático para extraer datos relevantes. El propósito del aprendizaje automático es aprender de los datos. Se han realizado muchos estudios sobre cómo hacer que las máquinas aprendan por sí mismas sin ser programadas explícitamente. Muchos matemáticos y programadores aplican varios enfoques para encontrar la solución a este problema que implica conjuntos de

datos enormes [13].

El Aprendizaje Automático se basa en diferentes algoritmos para resolver problemas de datos. A los científicos de datos les gusta señalar que no existe un único tipo de algoritmo que sea universal y mejor para resolver todos los problemas. El tipo de algoritmo utilizado depende del tipo de problema que se desee resolver, el número de variables, el tipo de modelo que se adapte mejor, entre otros factores.

Las técnicas de machine learning son necesarias para mejorar la precisión de los modelos predictivos. Dependiendo de la naturaleza del problema empresarial que se está atendiendo, existen diferentes enfoques basados en el tipo y volumen de los datos [14].

Una forma común de describir a un conjunto de datos u observaciones (dataset) es con una matriz de diseño. Una matriz de diseño es una matriz que contiene un ejemplo diferente en cada fila. Cada columna de la matriz corresponde a una característica diferente. Los algoritmos de aprendizaje se diferencian en función de la forma de operar y procesa los datos contenidos en una matriz de diseño [15].

En el contexto de la aplicación de algoritmos de Machine Learning en el cálculo de pronósticos de demanda, es esencial comprender que el objetivo principal de estos algoritmos es encontrar una función que tome un conjunto de variables como entrada y produzca una estimación del valor deseado, en este caso, la demanda. Esta función se ajusta a través del aprendizaje a partir de observaciones pasadas donde los datos de demanda son conocidos. Por ejemplo, utilizando datos de demanda de los últimos años, el modelo se entrena para desarrollar una función que pueda hacer predicciones precisas sobre la demanda futura, lo que implica una tarea de regresión, ya que el resultado esperado es un valor numérico real. Esta metodología se basa en el libro “Deep Learning” de Ian Goodfellow, Yoshua Bengio y Aaron Courville [16].

El siguiente diagrama muestra un flujo de trabajo típico para el uso del aprendizaje automático en modelado predictivo [17]:

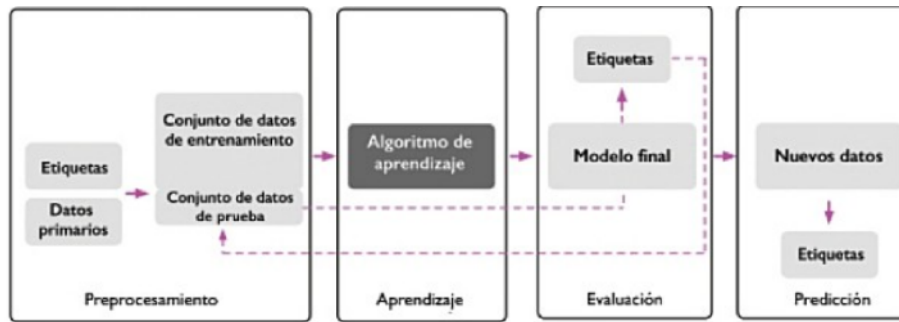


Figura 2.4: flujo de trabajo típico para el uso del aprendizaje automático en modelado predictivo

■ **Procesamiento:**

- Extracción y escalado de características.
- Selección de características.
- Reducción de la dimensionalidad.
- Muestreo.

■ **Aprendizaje:**

- Selección del modelo.
- Validación cruzada.
- Medición del rendimiento.
- Optimización de hiperparametro.

2.6.2. Aprendizaje supervisado

El aprendizaje supervisado comienza típicamente con un conjunto establecido de datos y una cierta comprensión de cómo se clasifican estos datos. El aprendizaje supervisado tiene la intención de encontrar patrones en datos que se pueden aplicar a un proceso de analítica. Estos datos tienen características etiquetadas que definen el significado de los datos. Por ejemplo, se puede crear una aplicación de machine learning con base en imágenes y descripciones escritas que distinga entre millones de animales [14].

El objetivo principal del aprendizaje supervisado es aprender un modelo, a partir de datos de entrenamiento etiquetados, que nos permite hacer predicciones sobre datos futuros o no vistos [17]. Aquí, el término supervisado

se refiere a un conjunto de muestras donde las señales de salida deseadas (etiquetas) ya se conocen.

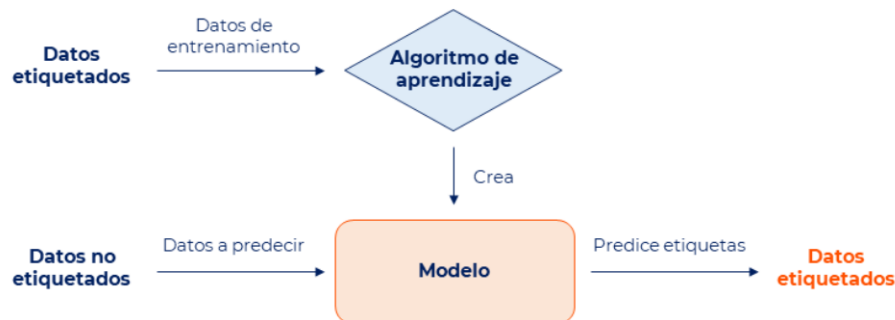


Figura 2.5: Aprendizaje supervisado.
[2]

2.6.3. Aprendizaje no supervisado

El aprendizaje no supervisado se utiliza cuando el problema requiere una cantidad masiva de datos sin etiquetar. Por ejemplo, las aplicaciones de redes sociales, tales como Twitter, Instagram y Snapchat, tienen grandes cantidades de datos sin etiquetar. La comprensión del significado detrás de estos datos requiere algoritmos que clasifican los datos con base en los patrones o clústeres que encuentra. El aprendizaje no supervisado lleva a cabo un proceso iterativo, analizando los datos sin intervención humana [14].

Los métodos de aprendizaje no supervisados son utilizados durante el preprocesamiento de los datos antes de ser utilizados por algoritmos de característica supervisada. Estos algoritmos también son muy utilizados en la compresión de datos. Todos los métodos son implícita o explícitamente basados en la distribución de probabilidad en el espacio definido por las variables de entrada [18].

2.6.4. Aprendizaje de refuerzo

El aprendizaje de refuerzo es un modelo de aprendizaje conductual. El algoritmo recibe retroalimentación del análisis de datos, conduciendo el usuario hacia el mejor resultado. El aprendizaje de refuerzo difiere de otros tipos de aprendizaje supervisado, porque el sistema no está entrenado con el conjunto de datos de ejemplo. Más bien, el sistema aprende a través de la prueba y el

error. Por lo tanto, una secuencia de decisiones exitosas conduce al fortalecimiento del proceso, porque es el que resuelve el problema de manera más efectiva [14].

2.6.5. Redes Neuronales Artificiales

Las Redes Neuronales Artificiales (RNA) son un componente esencial de la Inteligencia Artificial que emula el funcionamiento de las neuronas biológicas. Estas redes, entrenadas con entradas de escenarios internos o externos multiplicadas por pesos aleatorios, destacan en la resolución de funciones altamente no lineales, lo que las convierte en herramientas poderosas para la predicción. Inspiradas en el sistema nervioso biológico, encuentran aplicaciones en áreas como neurociencias, matemáticas, estadísticas y más. Su capacidad para aprender de datos de entrada las hace especialmente valiosas en la predicción de patrones complejos en diversos campos, como finanzas, ciencia de datos y análisis de mercado. Es un algoritmo basado en una red de alimentación de múltiples capas entrenadas inspirado en las neuronas del cerebro humano, estos sistemas aprenden y se forman a sí mismos ya que sus neuronas artificiales están conectadas, en lugar de ser programados de forma explícita [12].

En el desarrollo de una red neuronal no hay que programar ni el conocimiento ni las reglas del procesamiento del conocimiento. La red neuronal aprende las reglas del procesamiento del conocimiento mediante el ajuste de las conexiones ponderadas entre las neuronas de distintas capas de la red. El aprendizaje se consigue a través de una regla de aprendizaje que adapta o cambia los pesos de las conexiones en respuesta a los ejemplos de entrada, y opcionalmente también en respuesta a las salidas deseadas. Esta característica de las ANN es lo que permite decir que las redes neuronales aprenden de la experiencia [5].

La neurona artificial fue diseñada para “emular” las características del funcionamiento básico de la neurona biológica [19]. En esencia, se aplica un conjunto de entradas a la neurona, cada una de las cuales representa una salida de otra neurona. Cada entrada se multiplica por su “peso” o ponderación correspondiente análogo al grado de conexión de la sinapsis. Todas las entradas ponderadas se suman y se determina el nivel de excitación o activación de la neurona.

Representación vectorial del funcionamiento básico de una neurona artificial se indica según la siguiente expresión de la ecuación.

$$NET = X \cdot W$$

Siendo NET la salida, X el vector de entrada y W el vector de pesos.

La representación del modelo matemático es el siguiente

$$y = H \left(\sum_{j=1}^n w_j x_j - u \right) \quad (2.1)$$

Donde H es la función de activación (en este caso la función escalón de Heaviside) con el umbral u , x_j es la señal de entrada y w_j es el peso asociado con $j = 1, 2, \dots, n$, donde n corresponde al número de entradas. La salida de esta unidad es 1 cuando la suma está por encima del umbral u , y es 0 en caso contrario [15].

El elemento fundamental de los sistemas neuronales biológicos es la neurona, una célula viva que, como tal contiene todos los elementos que integran las células biológicas, si bien incorpora otros elementos que la diferencian. De forma genérica, una neurona consta de un cuerpo celular o soma más o menos esférico (de entre 10 y 80 micras de longitud), del que parten una rama principal o axón (cuya longitud varía desde las 100 micras hasta el metro en el caso de las neuronas motoras que constituyen los nervios) y un denso árbol de ramificaciones más cortas (árbol dendrítico), compuesto por dendritas (Figura 2.6). A su vez, el axón puede ramificarse en su punto de arranque, y con frecuencia presenta múltiples ramas en su extremo. La forma final de la neurona depende de la función que cumple, esto es, de la posición que ocupa en el conjunto del sistema y de los estímulos que recibe [20].

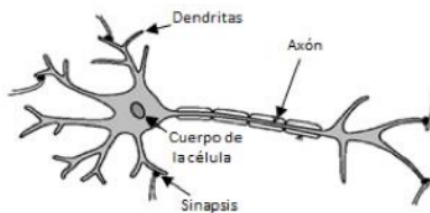


Figura 2.6: Componentes de una neurona.

2.6.6. Capas de la neurona artificial

Desde un punto de vista funcional, las neuronas constituyen procesadores de información sencillos, integrados por: Un canal de recepción de información, las dendritas, que reciben señales de entrada (inputs) procedentes de otras células (interneuronas) o del exterior (neuronas receptoras o sensoras, como los conos y bastones de la retina) [21].

Un órgano de cómputo, el soma, que combina e integra los inputs recibidos (generalmente a través de funcionales no lineales), emitiendo señales de salida en forma de estímulos nerviosos [21].

Un canal de salida, el axón, que envía la salida generada por el soma a otras neuronas o bien, en el caso de las neuronas motoras, directamente al músculo. Para transmitir la información, el axón se conecta a través de sus ramificaciones a las dendritas de otras neuronas, que reciben las señales y las combinan para producir nuevas salidas. Una neurona del córtex cerebral recibe información, por término medio, de unas 10.000 neuronas (convergencia), y envía impulsos a varios cientos de ellas (divergencia) [21].

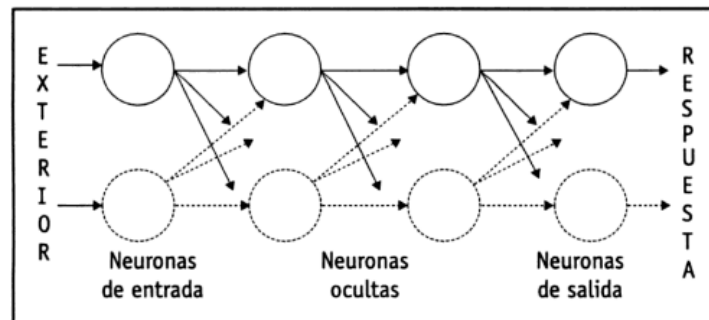


Figura 2.7: Tipos de neuronas artificiales.
[20]

Características de tres tipos de neuronas artificiales: unidades de entrada, de salida y unidades ocultas(Figura 2.7) [20].

- Las neuronas de entrada reciben señales desde el entorno, provenientes de sensores o de otros sectores del sistema (como archivos de almacenamiento de patrones de aprendizaje).
- Las neuronas de salida envían su señal directamente fuera del sistema una vez finalizado el tratamiento de la información (salidas de la red).

- Las neuronas ocultas reciben estímulos y emiten salidas dentro del sistema, sin mantener contacto alguno con el exterior. En ellas se lleva a cabo el procesamiento básico de la información, estableciendo la representación interna de ésta.

2.6.7. Modelo de redes neuronales artificiales

Las redes neuronales artificiales son motivadas por ciertas cualidades de su modelo real, por lo cual el desafío es producir un modelo que tenga [22]:

- Una estructura de procesamiento distribuida y paralela (opuestamente al CPU de una computadora).
- Alto grado de conexión entre las unidades básicas.
- Conexiones modificables en función de la experiencia.
- Un proceso de aprendizaje constante y de ser posible uno no supervisado.
- Aprendizaje basado en información local.
- Robustez en la performance si algunas unidades son removidas.

2.6.8. Arquitectura de una red neuronal

La topología o arquitectura hace referencia a la organización y disposición de las neuronas en la red formando capas de procesadores interconectados entre sí a través de sinapsis unidireccionales. La arquitectura de una RNA depende de cuatro parámetros principales [20]:

- el número de capas,
- el número de neuronas por capa,
- el grado de conectividad entre las neuronas y
- el tipo de conexiones neuronales.

Las arquitecturas neuronales pueden clasificarse atendiendo a distintos criterios [20]:

- Según su estructura en capas

- Redes monocapa, compuestas por una única capa de neuronas, entre las que se establecen conexiones laterales y, en ocasiones, autorrecurrentes. Este tipo de redes suele utilizarse para la resolución de problemas de autoasociación y clusterización.
 - Redes multicapa (layered networks), cuyas neuronas se organizan en varias capas (de entrada, oculta(s) y de salida). La capa a la que pertenece la neurona puede distinguirse mediante la observación del origen de las señales que recibe y el destino de la señal que genera.
- Según el flujo de datos en la red
- Redes unidireccionales o de propagación hacia adelante (feedforward), en las que ninguna salida neuronal es entrada de unidades de la misma capa o de capas precedentes. La información circula en un único sentido, desde las neuronas de entrada hacia las neuronas de salida de la red.
 - Redes de propagación hacia atrás (feedback), en las que las salidas de las neuronas pueden servir de entradas a unidades del mismo nivel (conexiones laterales) o de niveles previos. Las redes de propagación hacia atrás que presentan lazos cerrados se denominan sistemas recurrentes.

2.6.9. Perceptron

En 1957, Frank Rosenblatt publicó el mayor trabajo de investigación en computación neuronal realizado hasta esas fechas. Su trabajo consistía en el desarrollo de un elemento llamado “Perceptron” [5].

El perceptron es un sistema clasificador de patrones que puede identificar patrones geométricos y abstractos. El primer perceptron era capaz de aprender algo y era robusto, de forma que su comportamiento variaba sólo si resultaban dañados los componentes del sistema. Además presentaba la característica de ser flexible y comportarse correctamente después de que algunas celdas fueran destruidas. El perceptron fue originalmente diseñado para el reconocimiento óptico de patrones. Una rejilla de 400 fotocélulas, correspondientes a las neuronas de la retina sensibles a la luz, recibe el estímulo óptico. Estas fotocélulas están conectadas a elementos asociativos que recogen los impulsos eléctricos emitidos desde las fotocélulas. Las conexiones entre los elementos asociativos y las fotocélulas se realizan de forma aleatoria. Si las

células presentan un valor de entrada superior a un umbral predeterminado entonces el elemento asociativo produce una salida [5].

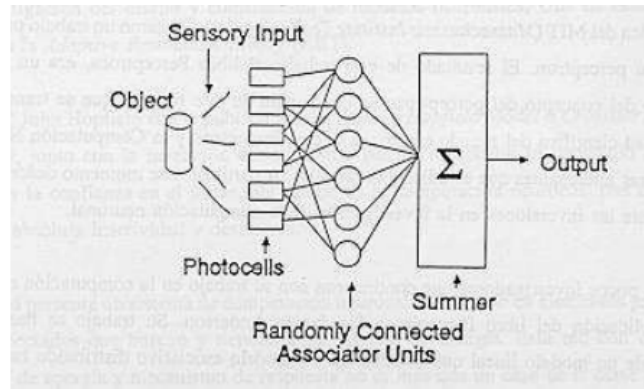


Figura 2.8: Aplicación de la Red Perceptron

2.6.10. Perceptrón multicapa

Una Perceptrón multicapa (MLP) puede ser interpretada como una extensión del algoritmo de regresión logística donde primero la entrada es transformada utilizando una transformación no lineal [18]. El propósito de esta transformación es proyectar los datos de entrada a un espacio donde sean linealmente separables. Esta capa intermedia es conocida como capa oculta y es característica de una MLP poseer dos o mas de ellas. Una capa oculta única es suficiente para hacer de una MLP un aproximador universal.

2.6.11. Redes neuronales recurrentes (RNN)

Una de las ideas principales que propició el desarrollo de estas redes fue extraída de las técnicas de aprendizaje automático y de los modelos estadísticos encontrados en la década de los 80. Esta idea se basaba en compartir diferentes parámetros a lo largo de diferentes partes de un modelo. Esto permitió generalizar conjuntos de datos que contaban con dependencias temporales entre sus entradas [23].

Un perceptrón multicapa tendría muy difícil el hecho de generalizar sobre una secuencia debido a que, si entrenamos esta red con secuencias de longitud fija, la red tendría parámetros separados para cada una de las características de entrada teniendo que aprender todas las reglas por separado. Sin embargo, la RNNs encontraría esta tarea mucho más fácil debido a que comparte pesos

a lo largo del tiempo. Cabe mencionar que cuando mencionamos el tiempo no tiene por que significar literalmente tiempo, puede simplemente significar la posición de un objeto en una secuencia [24].

Las redes neuronales recurrentes tienen caminos de retroalimentación entre todos los elementos que las conforman. Una sola neurona está entonces conectada a las neuronas posteriores en la siguiente capa, las neuronas pasadas de la capa anterior y a ella misma a través de vectores de pesos variables que sufren alteraciones en cada época con el fin de alcanzar los parámetros o metas de operación [25].

Las redes neuronales recurrentes (RNNs) son un tipo de modelo de aprendizaje profundo que se utiliza para analizar datos secuenciales, como el procesamiento del lenguaje natural o la predicción de series de tiempo [26].

A diferencia de las redes neuronales artificiales ya vistas, que asumen la independencia entre los datos de entrada, las RNN capturan activamente sus dependencias secuenciales y temporales [15].

Las RNN generalmente aumentan la arquitectura de red multicapa convencional con la adición de ciclos que conectan nodos adyacentes o pasos de tiempo. Estos ciclos constituyen la memoria interna de la red que se utiliza para evaluar las propiedades del dato actual con respecto a los datos del pasado inmediato. Las redes neuronales recurrentes son más eficaces para resolver problemas con no linealidades temporales significativas. Son especialmente útiles en aplicaciones tales como el reconocimiento de patrones secuenciales, cambiantes en el tiempo, ya que las capacidades de predicción y mapeo de las RNN así lo permiten [25].

Sin embargo, la capacidad de las RNNs para manejar datos a largo plazo se ve comprometida por el problema del gradiente que desaparece, que ocurre cuando el gradiente se hace cada vez más pequeño a medida que se propaga hacia atrás en la red. Para superar este problema, Hochreiter & Schmidhuber, propusieron la memoria a corto y largo plazo (LSTM), una variante de las RNNs que ha demostrado ser efectiva para el procesamiento de datos secuenciales a largo plazo [26].

2.6.12. Arquitectura básica de una RNN

Arquitectura básica de una RNN. Una característica importante es la inclusión de retrasos (z^{-1}) a la salida de las neuronas en las capas intermedias.

Las salidas parciales $S_{mn}(t+1)$ se convierten en valores $S_{mn}(t)$, un instante de tiempo anterior, y así se retroalimenta a todos los componentes de la red, guardando información de instantes de tiempo anteriores [25]. Se puede apreciar cómo todos los nodos están interconectados entre sí, estableciendo conexiones tanto directas como mediante retardos temporales con los nodos precedentes en cada capa, lo que permite la incorporación de memorias temporales.

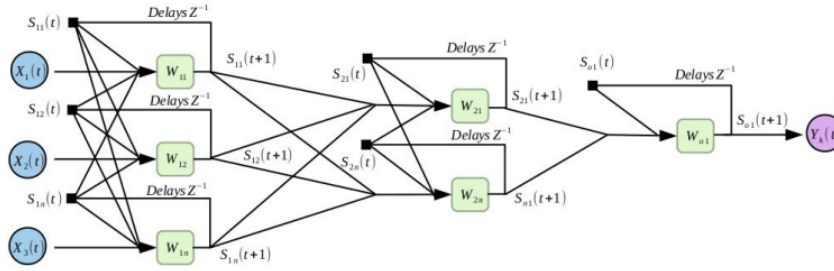


Figura 2.9: La arquitectura básica de una RNN

2.6.13. Redes de Memoria Corta y Larga LSTM

Las redes de “larga memoria de corto plazo” (Redes de Memoria Corta y Larga (LSTM)), propuestas por Hochreiter y Schmidhuber. Es una de las arquitecturas de aprendizaje profundo más avanzadas y exitosas para predicción de series temporales, reconocimiento de escritura y análisis de discurso [27].

El modelo matemático de LSTM se define como una función no lineal que transforma la entrada actual, el estado anterior y la memoria a largo plazo en una salida y un estado actualizado.

El cálculo de esta función implica la operación de multiplicación de matrices y la aplicación de funciones de activación, como la función sigmoide o la tangente hiperbólica [26].

Las unidades LSTM (Long Short-Term Memory) son unidades utilizadas en la construcción de redes recurrentes. Cada unidad LSTM es una celda con cuatro puertas (gates): la input gate, la external input gate, la forget gate y output gate. Lo más importante de la celda es su estado interno, encargado de recordar valores a lo largo del tiempo [24].

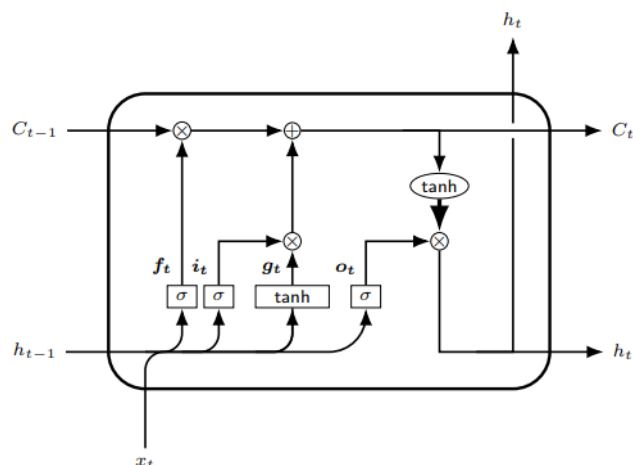


Figura 2.10: Representación de una celda LSTM.

El **estado interno** está representado en el diagrama anterior por C y recorre todas las celdas encadenadas, experimentando pequeños cambios cada vez que pasa por cada una de ellas.

El **estado oculto** (*hidden-state*) es, sin embargo, la salida de cada una de las unidades y se comparte a lo largo de todas las unidades de nuestro modelo. Está representado con la letra h.

El primer cambio que experimenta el estado interno es causado por la forget-gate (f_t). Esta puerta decide si el valor presente en el estado de la celda debe ser olvidado o no [24]. Esta decisión se toma aplicando la función sigmoide sobre la entrada y el estado oculto de la celda anterior, y multiplicando el resultado por el estado interno proveniente de la celda anterior. Dado que esta función toma valores entre 0 y 1, puede dar como resultado 0, lo que significa que se debe olvidar, o dar como resultado 1, lo que indica que el valor anterior debe ser completamente recordado.

$$f(t) = \sigma(b_f + U_f x(t) + W_f h(t-1))$$

2.6.14. Modelo de Regresión Lineal

La siguiente figura 2.11 ilustra el concepto de regresión lineal. Dada una variable predictora x y una variable de respuesta y, aplicamos una línea fina a este dato, que minimiza la distancia normalmente, la distancia cuadrada

de promedio entre los puntos de muestra y la línea aplicada. Ahora podemos utilizar la intersección y la pendiente aprendidas de este dato para predecir la variable de resultado del nuevo dato [17]:

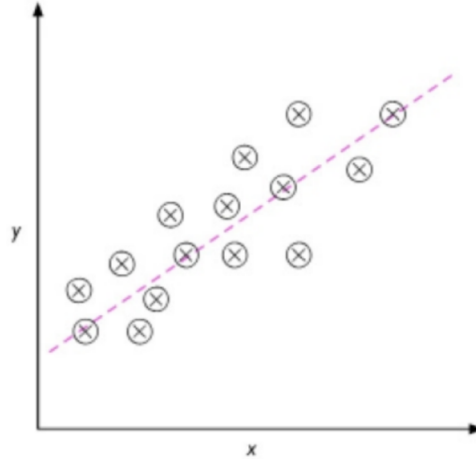


Figura 2.11: Regresión lineal

El modelo de regresión lineal es un método estadístico utilizado para modelar la relación entre una variable dependiente (la variable que queremos predecir) y una o más variables independientes (las variables que se utilizan para predecir la variable dependiente). El objetivo es encontrar una relación lineal que mejor se ajuste a los datos observados [28].

En general, si se incluyen cada vez más variables en un modelo de regresión, el ajuste a los datos mejora, aumenta la cantidad de parámetros a estimar pero disminuye su precisión individual (mayor varianza) y, por tanto, la de la función de regresión estimada, produciéndose un sobreajuste. Por el contrario, si se incluyen menos variables de las necesarias en el modelo, las varianzas se reducen, pero los sesgos aumentarán obteniéndose una mala descripción de los datos [29].

2.6.15. Los pesos sinápticos

A una neurona artificial se le asigna un peso sináptico a las entradas que provienen desde otras neuronas. Este procedimiento es similar al que se realiza en una neurona de un ser humano, a lo que normalmente en la medicina se le conoce como sinapsis. El peso sináptico entonces es un valor numérico y que puede ir cambiando durante la fase de entrenamiento [30].

Este peso hace que la red neural tenga una utilidad y es allí donde se almacena la información.

En una red de neuronas existe un peso o fuerza sináptica que va a ser un valor numérico que pondera las señales que se reciben por sus entradas. Este peso será un valor que determina la fuerza de conexión entre 2 neuronas. Cuando se evalúa una neurona se debe calcular el conjunto de todas las fuerzas o valores (denominado NET) que se reciben por sus entradas. Una vez calculado el valor conjunto de todas las entradas se aplica una función de activación (FA) que determinará el valor del estado interno de la neurona y que será lo que se transmita a su salida [31].

2.7. Métricas para la evaluación de una red neuronal

2.7.1. El error medio cuadrático (MSE)

El MSE es una medida del error cuadrático medio que se utiliza comúnmente para evaluar la precisión de un modelo de predicción [28]. Se define como la media de los errores cuadrados entre los valores observados y los valores predichos por el modelo. Esta métrica calcula el valor promedio de los errores elevados al cuadrado [32].

La expresión matemática es la siguiente:

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

Donde:

- n : número de observaciones.
- y_i : valor observado de la i -ésima observación.
- \hat{y}_i : valor predicho por el modelo para la i -ésima observación.

2.7.2. Raíz del error medio cuadrático (RMSE)

El RMSE (Error Cuadrático Medio de la Raíz) es una medida utilizada para evaluar la precisión de un modelo de predicción. Representa la raíz cuadrada de la media de los errores cuadrados entre los valores observados y los valores predichos por el modelo. Cuanto menor sea el valor de RMSE, mejor será el ajuste del modelo a los datos [28]. RMSE es una medida absoluta de

ajuste y se interpreta como la desviación estándar de la varianza inexplicada. Es especialmente útil cuando se busca evaluar la precisión de la predicción en el contexto de un modelo.

La expresión matemática es la siguiente:

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}$$

Donde:

- $RMSE$ es el Error Cuadrático Medio de la Raíz.
- n es el número de observaciones.
- y_i es el valor observado de la i -ésima observación.
- \hat{y}_i es el valor predicho por el modelo para la i -ésima observación.
- $\sqrt{}$ es la función que calcula la raíz cuadrada.
- \sum es la función que suma los términos.

2.7.3. Error Absoluto Medio (MAE)

El margen de error que se produce con la predicción se denomina error absoluto medio (MAE, por sus siglas en inglés), métrica utilizada para ver el rendimiento en las predicciones y que da cuenta de la diferencia promedio entre las estimaciones obtenidas por los clasificadores utilizados y los resultados reales. Mientras más próximo sea el MAE a 0, más cercano al resultado real de las elecciones [33].

Es una medida que calcula la diferencia promedio absoluta entre los valores pronosticados y los valores reales en un pronóstico [32]. El MAE es útil para evaluar cuán cerca están los pronósticos de los valores reales. Su fórmula es la siguiente:

$$MAE = \frac{SAE}{N} = \frac{\sum_{i=1}^N |y_i - \hat{y}_i|}{N}$$

Donde:

- MAE es el Error Absoluto Medio.
- N es el número de puntos de datos no faltantes.

- y_i corresponde a la información actual de la serie de tiempo.
- \hat{y}_i corresponde a la serie de tiempo pronosticada.
- \sum es la función que realiza la suma.
- SAE Corresponde a la sumatoria de errores absolutos o desviaciones
- $||$ representa el valor absoluto.

2.7.4. Error porcentual absoluto medio (MAPE)

Permite calcular la dimensión del error de tipo absoluto expresado en porcentajes, y su fórmula es la siguiente [32]:

$$MAPE = \frac{1}{N} \sum_{t=1}^N \frac{|X_t - Y_t|}{|X_t|}$$

Donde:

- X_t : Valor real.
- Y_t : Valor de pronóstico.
- N : Número de puntos de datos no faltantes.

2.7.5. El coeficiente de determinación R^2

Es una medida que resume la capacidad explicativa de un modelo de regresión. Describe la proporción de la varianza en la variable dependiente que puede ser explicada por el modelo de regresión [32]. Su fórmula es:

$$R^2 = \frac{SSR}{SST} = 1 - \frac{SSE}{SST}$$

De lo cual:

- SST : Corresponde a la Suma de Cuadrados Total.
- SSR : Corresponde a la Suma de Cuadrados de Regresión.
- SSE : Corresponde a la Suma de Cuadrados de Error.

2.8. Antecedentes

En el trabajo titulado ***“Optimización de la cadena de abastecimiento a través de un sistema inteligente de pronósticos de demanda y gestión de inventario multiproducto”*** [34]. El objetivo principal del proyecto fue desarrollar una aplicación web que automatizara parcialmente las actividades del área de compras y mejorara la cadena de abastecimiento mediante un sistema inteligente de pronósticos de demanda y gestión de inventario multiproducto. Este antecedente se presenta como una sólida prueba de concepto para la implementación de sistemas inteligentes de gestión de compras basados en pronósticos de demanda en la industria gastronómica, lo que respalda y valida la relevancia de la presente tesis sobre un sistema de compra inteligente basado en historial de ventas.

Trabajo final de grado titulado ***“Estudios de predicción en series temporales de datos meteorológicos utilizando redes neuronales recurrentes”*** [25] En los últimos años, en el campo de las energías renovables, la energía eólica ha sido una de las que mas se ha desarrollado e invertido. La importancia de las predicciones de viento radica en la ayuda que aportan para planificar y anticiparse a los valores futuros que afectarán al sistema, ayudando a gestionar la adquisición de los recursos necesarios con antelación suficiente. Recientemente se han desarrollado nuevas arquitecturas de redes recurrentes que resultan muy prometedoras para realizar predicción. En este trabajo se probará y experimentará con dichas arquitecturas para realizar distintas predicciones de la velocidad del viento en un horizonte de corto y muy corto plazo a partir de datos de series temporales de viento.

En el proyecto titulado ***“Desarrollo de un Sistema de Control de Inventario para la Gestión de Compras de Materia Prima en el Rubro de Restaurantes”*** [35]. Este antecedente presenta un estudio de caso relevante en el ámbito de la gestión de inventario y compras en la industria de restaurantes. Se describe cómo se implementó con éxito un sistema de control de inventario utilizando el modelo de desarrollo de ciclo de vida en cascada. El objetivo principal de esta implementación fue mejorar la gestión de los procesos de almacén y reducir los tiempos innecesarios en la entrega de productos a los clientes. El sistema se dividió en módulos específicos, centrándose en el almacén y ofreciendo diversas funcionalidades para la gestión eficiente del restaurante. Los resultados obtenidos demuestran una modernización efectiva de los procesos de la empresa en el rubro de restaurantes, con mejoras significativas en la gestión y una reducción sustancial de

los tiempos de almacenamiento. Este antecedente sirve como base para la presente tesis, que se enfoca en el desarrollo de un sistema de compra inteligente basado en el historial de ventas, aprovechando la experiencia exitosa de la implementación previa para impulsar la eficiencia operativa en la industria de alimentos

Tesis ***“Predicción de la demanda usando modelos de machine Learning”*** [36] Para aquellas empresas dedicadas a la venta en retail o venta directa donde su portafolio de productos es muy amplio, la planeación de la demanda se convierte en un área determinante para la correcta administración del flujo de caja, rentabilidad y efectividad en ventas por varias razones: la primera de ellas es la gestión de compra de insumos por medio de negociación de precio por volumen con sus proveedores; el control de inventario donde se cuida un equilibrio entre uso efectivo del espacio de almacenamiento y reducción de obsolescencia contra la disponibilidad para distribución y por último en la venta efectiva respetando las estacionalidades, tendencias del mercado y satisfacción del cliente.

El artículo ***“Reducing Food Waste in the Food Industry with Deep Learning”*** [37].

El autor, Esteban David Romero Pérez, se centra en la importante tarea de ayudar a la industria alimentaria a anticipar con precisión la demanda de sus productos y, por lo tanto, reducir los excedentes no vendidos, lo que está alineado con el Objetivo de Desarrollo Sostenible número 12 de las Naciones Unidas. Los resultados obtenidos en este estudio demuestran de manera convincente que la aplicación de Deep Learning puede conducir a una disminución sustancial en la cantidad de alimentos desperdiciados en la industria alimentaria, contribuyendo significativamente a un futuro más sostenible y eficiente en el uso de recursos. Este antecedente resalta la relevancia de la tecnología de aprendizaje profundo en la optimización de procesos en la industria alimentaria, lo que puede ser de gran utilidad para el enfoque de la presente tesis.

Capítulo 3

Método, enfoque e implementación

3.1. Método

La investigación se centra en el desarrollo de un sistema de compras inteligentes que utiliza datos históricos de ventas para optimizar la gestión de insumos gastronómicos. La metodología implica la recopilación y análisis de datos de ventas previas para identificar patrones y la aplicación de algoritmos de aprendizaje automático para predecir futuras compras. Se evaluará la precisión del sistema mediante métricas cuantitativas, y el alcance de la investigación se concentra en un local gastronómico en Ciudad del Este, detallando la recopilación de datos, la arquitectura del sistema y su capacidad para generar predicciones de demanda basadas en el histórico de ventas.

3.2. Preprocesamiento de datos

En esta sección, se describirán los métodos utilizados para llevar a cabo la investigación sobre la predicción de compras a partir de un historial de ventas.

3.2.1. Obtención de datos

Los datos usados para generar el modelo fueron proporcionados por la empresa gastronómica. La misma proporcionó el dataset para ser usado con fines académicos, los datos presentan las ventas diarias para los diferentes productos, cuenta con 6 meses de registro desde el 17 de abril del 2023 hasta el 15 de octubre del 2023.

Los datos originales suministrados están conformados por los siguientes datasets:

- productos.csv
- ventas.csv

Los datos vienen organizados en forma tabular, cada archivo presenta información separada relacionada con los diferentes productos y el registro de ventas histórico.

productos.csv

Estructura inicial de la tabla de productos (información de los 5 primeros productos(Fig. 3.3)).

id	codigo	id_categoria	producto	marca	descripcion	precio_costo	precio_minorista
1	992856	1	Simple	1		0	14000
2	815013	1	Hamburguesa doble	1		0	22000
3	560531	1	Simple	1		0	18000
4	446168	2	Hamburguesa doble bacon	1		0	27000
5	864790	2	Papa pequeña	1		0	10000

Figura 3.1: Estructura inicial de la tabla productos.csv

ventas.csv

Estructura inicial de la tabla de ventas(con información de 6 registros (Fig. 3.2)).

id	id_presupuesto	id_cliente	id_vendedor	id_producto	observacion	precio_venta	cantidad	fecha_presupuesto
2	2	0	37	69	NULL	28000	1	2023-04-17 18:38:00
3	2	0	37	59	NULL	22000	2	2023-04-17 18:38:00
4	2	0	37	54	NULL	17000	1	2023-04-17 18:38:00
5	3	0	37	91	NULL	13000	1	2023-04-17 18:43:00
6	3	0	37	95	NULL	17000	1	2023-04-17 18:43:00
7	3	0	37	94	NULL	11000	1	2023-04-17 18:43:00

Figura 3.2: Estructura inicial de la tabla ventas.csv

3.2.2. Modelado de datos

Antes de evaluar algún modelo, se debe llevar a cabo una limpieza de la base de datos y acondicionamiento de las características finales en variables numéricas.

En el siguiente apartado se estará realizando el proceso de limpieza de los datos para la obtención de los parámetros a utilizar.

Se obtuvieron las columnas necesarias de la tabla productos para el desarrollo del proyecto, desde la base de datos MySQL herramienta utilizada para el almacenamiento de los datos.


Hay varios sistemas de gestión de bases de datos gratuitos o de bajo coste disponibles, como MySQL, PostgreSQL o SQLite. Cuando comparas MySQL con otros sistemas de bases de datos, piense en lo que es más importante para usted. Características de rendimiento (como compatibilidad o extensiones de SQL), soporte, condiciones de licencia y precio, todo son factores a tener en cuenta [38].

Dadas estas consideraciones, MySQL tiene muchas cualidades atractivas:

- Velocidad.
- Facilidad de uso.
- Soporte de lenguaje de consulta. MySQL entiende SQL (consulta estructurada Language), el lenguaje estándar elegido para todos los sistemas de bases de datos modernos.
- Capacidad. El servidor MySQL es multihilo, lo que permite que muchos clientes se conecten a él al mismo tiempo. Cada cliente puede usar múltiples bases de datos simultáneamente.
- Conectividad y seguridad. MySQL es completamente en red, y las bases de datos se pueden acceder desde cualquier lugar de Internet, lo que le permite compartir sus datos con cualquier persona, en cualquier lugar.
- Portabilidad. MySQL se ejecuta en muchas variedades de Unix y Linux, así como en otros sistemas como Windows.
- Disponibilidad y costos. MySQL es un proyecto de código abierto disponible bajo múltiples términos de licencia

La siguiente consulta SQL selecciona las columnas `id` y `producto` de la tabla `productos`, donde la columna `anulado` tiene un valor nulo:

```
SELECT id, producto
FROM productos
WHERE anulado IS NULL;
```



id	producto
1	Hamburguesa Simple
2	Hamburguesa doble
3	Hamburguesa triple
4	Hamburguesa doble bacon
5	Papa pequeña

Figura 3.3: Datos de la tabla `productos`(Los primeros 5 productos)

En la tabla 3.1 se muestra una descripción de cada una de las columnas de productos.

Columna	Descripción
id	el id del producto (identificador unico)
producto	el nombre del producto

Tabla 3.1: Descripción de datos de la tabla de productos

La consulta SQL a continuación selecciona las columnas `id_producto`, `cantidad`, y `fecha_venta` de la tabla `ventas` para las transacciones que tuvieron lugar en el período desde el 17 de abril de 2023 hasta el 15 de octubre de 2023 donde la columna `anulado` tiene un valor nulo (Figura:3.4):

```
SELECT id_producto, cantidad, fecha_venta
FROM ventas
WHERE fecha_venta BETWEEN '2023-04-17' AND '2023-10-15'
AND anulado IS NUL;
```

id_producto	cantidad	fecha_venta
45	12	2023-04-17 17:58:00
37	1	2023-04-17 17:58:00
13	1	2023-04-17 17:58:00
36	1	2023-04-17 17:58:00
26	1	2023-04-17 17:58:00
9	1	2023-04-17 17:58:00
66	1	2023-04-17 17:59:00

Figura 3.4: Datos de la tabla ventas

En la tabla 3.2 se muestra una descripción de cada una de las columnas de productos.

Columna	Descripción
id_producto	el id del producto (identificador unico)
catidad	cantidad de venta
fecha_venta	fecha de la venta

Tabla 3.2: Descripción de datos de la tabla ventas

3.2.3. Escalado de datos

El escalado de datos es un proceso esencial en el análisis de datos y la estadística que ajusta los valores de las variables para que compartan una misma escala o propiedades específicas.

La consulta SQL se enfoca en el escalado de datos de ventas de productos durante un periodo del 17 de abril al 15 de octubre, excluyendo ventas anuladas. La consulta agrupa los resultados por producto y fecha de venta, permitiendo así obtener una visión escalada de la cantidad total vendida de cada producto a lo largo del tiempo, lo que facilita la identificación de tendencias y patrones en las ventas.

```
SELECT id_producto, SUM(cantidad) AS Cantidad, fecha_venta
FROM ventas
WHERE fecha_venta BETWEEN '2023-04-17' AND '2023-10-15'
AND anulado IS NULL
```

GROUP BY id_producto, fecha_venta;

id_producto	Cantidad	fecha_venta
49	13	2023-04-17
54	13	2023-04-17
59	3	2023-04-17
69	4	2023-04-17
75	1	2023-04-17
87	3	2023-04-17
91	1	2023-04-17
92	2	2023-04-17
93	5	2023-04-17

Figura 3.5: Datos escalados(muestra de la tabla ventas)

Columna	Descripción
id_producto	el id del producto
cantidad	suma de la cantidad de ventas por fecha y producto
fecha_venta	fecha de la venta

Tabla 3.3: Descripción de datos escalados de la tabla ventas

3.2.4. Análisis de los datos

El objetivo principal de esta etapa es explorar y comprender en profundidad los datos, lo que resulta fundamental para una sólida preparación y análisis de los mismos. Esta comprensión más profunda del problema de negocio nos permitirá seleccionar modelos de predicción y tomar decisiones más fundamentadas. Durante el proceso de limpieza y preparación de los datos, se han identificado características relevantes que proporcionarán información valiosa para las fases posteriores del análisis y la toma de decisiones.

Como se observa(Figura: 3.4), la base de datos de entrenamiento es limitada en términos de variables disponibles. En este escenario, se ha seleccionado exclusivamente la fecha como variable de entrada y se ha focalizado en un producto específico, en este caso, la “Hamburguesa simple” con id_producto

49, debido a su alta demanda. Se han recopilado 182 registros ya agrupados por día, lo que equivale a 182 días de datos. El propósito de este enfoque es analizar el comportamiento de las ventas de dicho producto en relación al tiempo y buscar patrones que puedan mejorar la precisión del modelo de predicción.

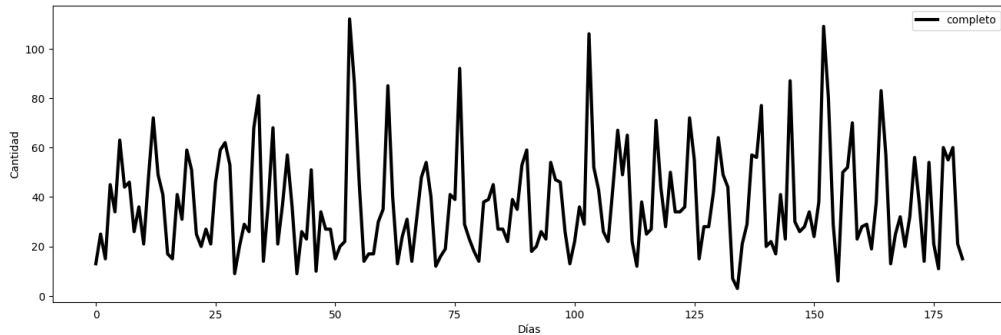


Figura 3.6: Serie temporal de las ventas totales por día

3.2.5. Definición de las variables de entrada y salida

Se han considerado estas variables como variables de entrada debido a que, a través del análisis de la serie temporal de ventas, se han identificado patrones y tendencias significativas que impactan en las ventas del producto.

El mes, el día del mes, el día de la semana, la presencia de días festivos y la estación del año influyen directamente en el comportamiento de las ventas.

El promedio de venta mensual varía, lo que indica una estacionalidad en la demanda. Las ventas tienden a aumentar hacia el final e inicio del mes, posiblemente relacionado con el ciclo de pagos de los consumidores. La influencia de los días de la semana sugiere que los fines de semana son momentos clave para las ventas. La presencia de días festivos genera picos en las ventas, lo que puede estar vinculado a la celebración y al aumento de la demanda en esas fechas. Por último, la estación del año también desencadena cambios en las ventas, con un aumento en verano, lo que respalda la necesidad de considerar esta variable en el modelo para una predicción más precisa.

Teniendo todas estas informaciones, se procede a la conversión de estas características en variables de entrada y salida para el aprendizaje del modelo, y se asegura de que estén representadas en formato numérico.

Columna	Descripción	Dato numerico
mes	variable mes Rango	1-12
dia	variable dias del mes Rango	1-31
dia_semana	variable dia de la semana Rango	0-6
dia_festivo	variable dia festivo Rango	0-1
estacion	variable estaciones del año Rango	0-3

Tabla 3.4: Variables de entrada

La siguiente consulta SQL realiza un análisis de ventas de un producto con `id_producto` igual a 49. Agrupa los datos por fecha de venta y calcula diversas métricas relacionadas con la fecha, como el mes, el día, el día de la semana y si la fecha es un día festivo. También determina la estación correspondiente a cada fecha y agrega información sobre la cantidad total de ventas y la suma de las cantidades vendidas para cada día. En resumen, esta consulta proporciona un resumen detallado de las ventas del producto 49, organizado por fecha y con métricas relacionadas con la fecha para su posterior utilización.

```
SELECT MONTH(fecha_venta) AS mes,  
DAY(fecha_venta) AS dia,  
DAYOFWEEK(fecha_venta) AS dia_semana,  
IFNULL((SELECT 1 FROM dias_festivos df  
WHERE CAST(v.fecha_venta AS date)=df.fecha),0) AS dia_festivo,  
(SELECT estacion FROM estaciones e  
WHERE cast(v.fecha_venta AS date)  
BETWEEN e.fecha_inicio AND e.fecha_fin) AS estacion,  
COUNT(id_venta) AS cantidad_ventas,  
SUM(v.cantidad) AS cantidad  
FROM ventas v  
WHERE id_producto = 49  
GROUP BY CAST(fecha_venta AS date)
```


mes	dia	dia_semana	dia_festivo	estacion	cantidad
4	17	2	0	2	13
4	18	3	0	2	25
4	19	4	0	2	15
4	20	5	0	2	45
4	21	6	0	2	34
4	22	7	0	2	63
4	23	1	0	2	44

Figura 3.7: Tabla resultante con las variables de entrada

3.2.6. Lectura y escalado del dataset en Python

La elección de Python como lenguaje de programación se fundamenta en su destacado uso en el campo de la inteligencia artificial, respaldado por una amplia gama de bibliotecas y una comunidad activa [17]. En cuanto al entorno de pruebas, se ha optado por Google Colab debido a su facilidad de uso y su gratuidad, lo que lo convierte en una opción conveniente para el desarrollo y la prueba de modelos de inteligencia artificial.

Los datos se preparan de manera similar tanto para las pruebas en el modelo de regresión lineal como para el modelo LSTM, lo que implica el proceso de escalado y la división en variables de entrada y salida (Figura: 3.8).

Python es un lenguaje muy maduro, versátil, y se puede utilizar como una plataforma específica para el Data Science, gracias a su gran ecosistema de librerías científicas y a su comunidad, que es muy activa [39].

En este proyecto se emplean algunas de ellas que nos facilitaran el proceso a la hora de realizar diferentes funciones.

- **NumPy:** Ofrece una amplia gama de funciones matemáticas y herramientas para trabajar con datos numéricos, como generadores de números aleatorios, rutinas de álgebra lineal, transformadas de Fourier y más. Esto hace que NumPy sea una biblioteca esencial en aplicaciones científicas y de análisis de datos, ya que proporciona las herramientas necesarias para realizar cálculos numéricos de manera eficiente y rápida [40].

- **Pandas:** Es una biblioteca de Python ampliamente utilizada en la manipulación y análisis de datos. Ofrece un objeto eficiente llamado DataFrame para trabajar con datos tabulares, facilitando la lectura y escritura de datos desde diversas fuentes [41].
- **Matplotlib:** Esta librería de software gráfico permite la generación de gráficos en dos dimensiones, a partir de datos contenidos en listas o arrays. Junto con Pandas han sido las dos librerías de Python que más se han empleado en el proyecto para el tratamiento de los datos [42].
- **TensorFlow:** Es una potente biblioteca de aprendizaje automático que permite a los desarrolladores y científicos de datos diseñar, entrenar y desplegar modelos de inteligencia artificial en una amplia gama de aplicaciones. [43].
- **Keras:** Es una biblioteca de alto nivel para construir y entrenar redes neuronales en Python. Ofrece una interfaz sencilla y modular, siendo valiosa para desarrolladores y científicos de datos. Aunque es compatible con varios backends, TensorFlow se utiliza comúnmente en combinación con Keras para proyectos de aprendizaje profundo [44].
- **Scikit-Learn:** Es una librería de aprendizaje automático que contiene herramientas para la predicción de series temporales, incluyendo modelos de regresión y redes neuronales [28].

Además de estas librerías, existen otras herramientas que se pueden utilizar en Python para la predicción de series temporales, como Prophet, que es una librería para el análisis y la predicción de series temporales con estacionalidad.

```
import numpy as np
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error, mean_absolute_error
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
import matplotlib.pyplot as plt
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense

# Cargar el conjunto de datos
data = pd.read_csv('dataset12.csv')

# Elegir las características y el objetivo
features = ['mes', 'dia', 'dia_semana', 'dia_festivo', 'estacion', 'cantidad']
target = 'cantidad'

X = data[features].values
Y = data[target].values

# Normalizar los datos
scaler_X = MinMaxScaler()
scaler_Y = MinMaxScaler()

X_scaled = scaler_X.fit_transform(X)
Y_scaled = scaler_Y.fit_transform(Y.reshape(-1, 1))
```

Figura 3.8: Código de lectura y escalado del dataset en Python

El gráfico de violín es una herramienta valiosa en el análisis de datos y, en particular, en el contexto de escalado de datos, ya que proporciona una representación más completa y detallada de las distribuciones, lo que puede ayudar a comprender mejor la estructura de los datos y evaluar los efectos del escalado en la distribución de datos.

En el gráfico 3.9 se puede observar que todas las variables de entrada están escaladas entre 0 y 1 incluyendo el set train, validación y test.

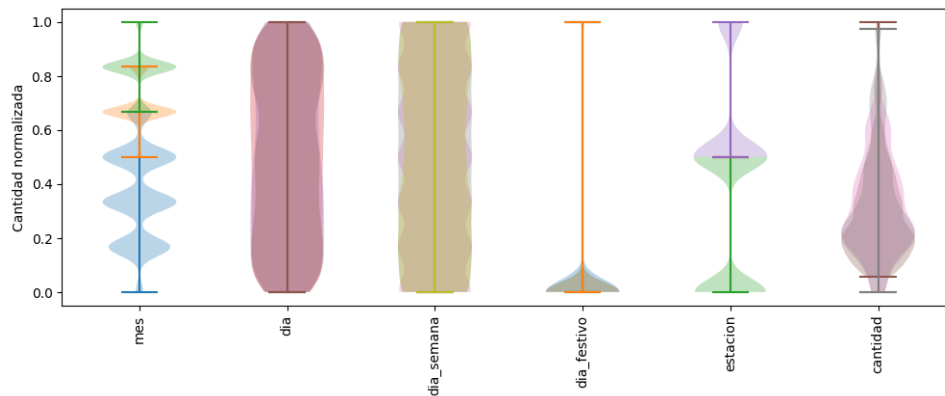


Figura 3.9: Grafico en violín incluyendo los tres set train, validación y test

En el grafico 3.10 se puede observar que los datos de salida también estén escalados correctamente.

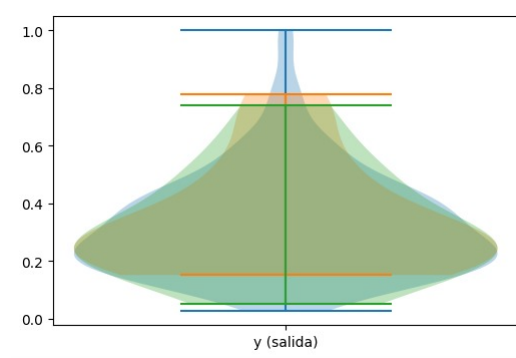


Figura 3.10: Gráfico en violín de los datos de salida escaladas

3.2.7. Distribución del conjunto de datos

En el proceso de entrenamiento, es fundamental dividir el conjunto de datos en tres conjuntos distintos:

- **Conjunto de Entrenamiento (Train):** Este conjunto se utiliza para entrenar el modelo, lo que significa que el modelo aprenderá de estos datos durante el proceso de entrenamiento.
- **Conjunto de Validación (Val):** El conjunto de validación se emplea para evaluar si el modelo está sobreajustando los datos de entrenamiento. Ayuda a ajustar parámetros y prevenir el sobreajuste.

- **Conjunto de Prueba (Test):** Este conjunto no se utiliza durante el entrenamiento del modelo y contiene datos con valores desconocidos. Se utiliza al final para evaluar la precisión del modelo en un entorno real, ya que no ha tenido acceso a estos datos previamente.

Se toma inicialmente 70 % train, 15 % val y 15 % test, como valores comúnmente utilizados para estos casos.

```
# Dividir los datos en conjuntos de entrenamiento, validación y prueba
train_size = int(0.7 * len(X_seq))
val_size = int(0.15 * len(X_seq))
test_size = len(X_seq) - train_size - val_size

X_train, X_val, X_test = X_seq[:train_size], X_seq[train_size:train_size + val_size], X_seq[train_size + val_size:]
Y_train, Y_val, Y_test = Y_seq[:train_size], Y_seq[train_size:train_size + val_size], Y_seq[train_size + val_size:]

print('Tamaños entrada (BATCHES x INPUT_LENGTH x FEATURES) y de salida (BATCHES x OUTPUT_LENGTH x FEATURES)')
print(f'Set de entrenamiento - x_tr: {X_train.shape}, y_tr: {Y_train.shape}')
print(f'Set de validación - x_vl: {X_val.shape}, y_vl: {Y_val.shape}')
print(f'Set de prueba - x_ts: {X_test.shape}, y_ts: {Y_test.shape}')
```

Tamaños entrada (BATCHES x INPUT_LENGTH x FEATURES) y de salida (BATCHES x OUTPUT_LENGTH x FEATURES)
Set de entrenamiento - x_tr: (107, 23, 6), y_tr: (107, 7, 1)
Set de validación - x_vl: (22, 23, 6), y_vl: (22, 7, 1)
Set de prueba - x_ts: (24, 23, 6), y_ts: (24, 7, 1)

Figura 3.11: Código de distribución de los datos.

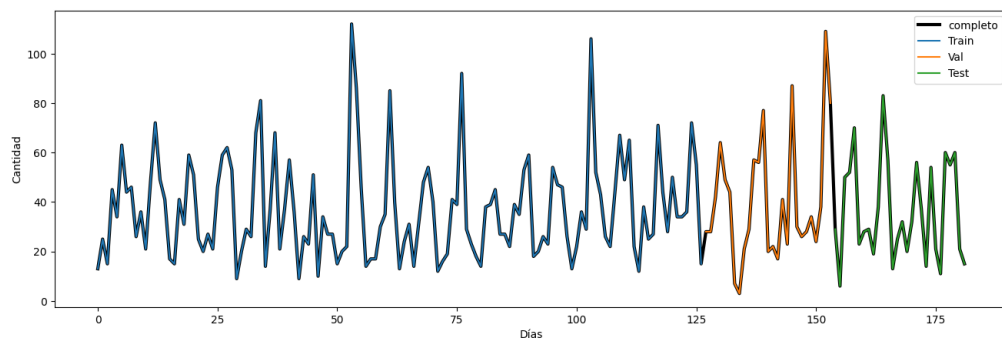


Figura 3.12: Distribución de los datos, ventas totales por día.

3.3. Modelos de predicción

Finalmente, todo el desarrollo del trabajo que se describirá a continuación referente al tratamiento de datos, entrenamiento de los modelos, evaluación de las métricas y su representación gráfica se ejecutará usando Python con

el fin de aprovechar su uso por medio de librerías como Numpy, Matplotlib, Pandas, Scikit-Learn el cual facilitará la solución de los requerimientos necesarios en cada punto del proceso.

3.3.1. Modelo de Regresión lineal

Para la implementación de regresión lineal, se ha optado por utilizar el SGDRegressor de la biblioteca sklearn. En lo que respecta a la elección de hiperparámetros, se han seleccionado valores comunes para un modelo de regresión lineal implementado a través del algoritmo de Descenso de Gradiente Estocástico (SGD).

Estos parametros son los siguientes:

- **loss**: Es una función de pérdida en la regresión lineal que mide la diferencia entre las predicciones del modelo y los valores reales. Su objetivo es minimizar esta discrepancia para encontrar la mejor línea de ajuste a los datos.
- **penalty**: Se ha especificado la regularización Ridge, que es una técnica comúnmente utilizada para abordar problemas de regresión. La regularización Ridge ayuda a prevenir el sobreajuste incorporando un término de penalización en la función de pérdida.
- **alpha**: El hiperparámetro de regularización controla la intensidad de la regularización.
- **max_iter**: Se ha establecido el número máximo de iteraciones permitidas para el descenso de gradiente.
- **tol**: La tolerancia se utiliza para determinar cuándo se ha alcanzado la convergencia.
- **learning_rate**: Se ha configurado la tasa de aprendizaje como constante a lo largo del proceso de entrenamiento.

Fase de entrenamiento:

En esta fase se tiene una cantidad enorme de datos, de la cual se separa una parte para entrenar al algoritmo y darle toda esta información para que encuentre los patrones necesarios y después pueda hacer predicciones.

El modelo se entrena con los datos de entrenamiento (X_train, Y_train) utilizando el método .fit(). El modelo ajusta la línea de regresión a estos datos para realizar predicciones (Figura 3.13).

Fase de prueba:

El resto de los datos que quedan, se van a usar para hacer las pruebas. Así le podemos hacer preguntas al algoritmo y evaluar si las respuestas están bien o mal, y saber si está aprendiendo o no. Si vemos que no coinciden los datos, tendremos que agregar más datos o cambiar el método que estamos utilizando. Pero si se observa que hay entre un 80 % a 90 % de respuestas correctas, podemos decir que hay un buen grado de aprendizaje y poder utilizar ese algoritmo.

Realizar Predicciones en Conjuntos de Datos: (Figura 3.13)

- Se realizan predicciones en tres conjuntos de datos diferentes: entrenamiento, validación y prueba.
- Para cada conjunto, se utiliza el modelo para predecir los valores de la variable objetivo.
- Las predicciones se almacenan en las variables $Y_{\text{train_pred}}$, $Y_{\text{val_pred}}$ y $Y_{\text{test_pred}}$, respectivamente.
- Se utiliza el método `.ravel()` para asegurarse de que las predicciones tengan la forma correcta.

```
# Crea el modelo de regresión lineal con SGD
model = SGDRegressor(
    loss=loss, # Hiperparámetro pérdida
    penalty=penalty, # Hiperparámetro de penalidad
    alpha=alpha, # Hiperparámetro de regularización
    max_iter=max_iter, # Número máximo de iteraciones
    tol=tol, # Tolerancia para la convergencia
    learning_rate=learning_rate, #Tasa de aprendizaje
    eta0=0.01 # Tasa de aprendizaje inicial
)
model.fit(X_train, Y_train)

# Realizar predicciones en el conjunto de prueba
Y_train_pred = model.predict(X_train)
Y_train_pred = Y_train_pred.ravel()
Y_val_pred = model.predict(X_val)
Y_val_pred = Y_val_pred.ravel()
Y_test_pred = model.predict(X_test)
Y_test_pred = Y_test_pred.ravel()
```

Figura 3.13: Código de la estructura de regresión lineal.

Fase de validacion:

El proceso de validación implica calcular y comparar estas métricas utilizando los datos de validación y prueba. Un buen modelo de regresión lineal debería tener valores bajos de MSE y MAE, así como un alto valor de R^2 en ambos conjuntos de datos, lo que indica que las predicciones se ajustan bien a los valores reales. La validación es esencial para evaluar el rendimiento del modelo en datos no vistos y garantizar su capacidad de generalización.

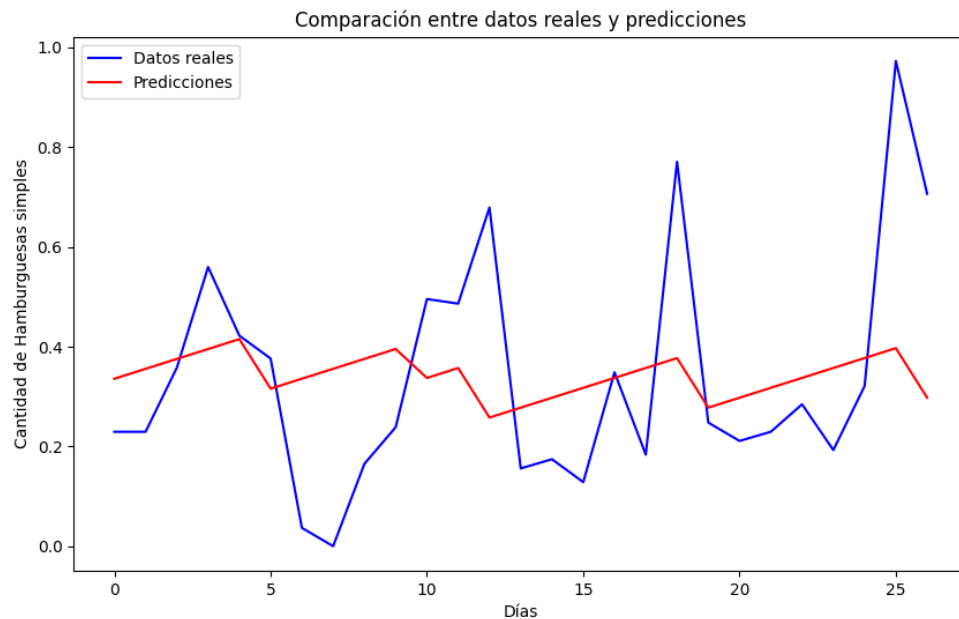


Figura 3.14: Grafico de lineas con Matplotlib.

3.3.2. Modelo LSTM

La red neuronal LSTM se modela con ayuda de la librería keras, es una librería de código abierto de redes neuronales, que se implementa sobre otros frameworks de aprendizaje automático como Tensorflow, CNTK o theano. Apareció en 2015 y fue creada por François Chollet. Tal y como se menciona en su documentación [44], Keras se guía por cuatro principios. Ser amigable para el usuario, modular, fácilmente extensible e implementada en python.

Al configurar una red neuronal LSTM, es común considerar varios hiperparámetros clave (Codigo 4.5):

- **Unidades de LSTM:** Por lo general, se comienza con un número moderado de unidades en las capas LSTM, como 64 o 128. Aumentar el número de unidades puede ayudar a capturar patrones más complejos, aunque conlleva un mayor riesgo de sobreajuste.
- **Número de capas LSTM:** Para problemas más complejos, es posible agregar capas adicionales. Las arquitecturas con dos capas LSTM son comunes y funcionan bien en muchas aplicaciones.

- **Función de Activación:** Para las capas LSTM, la función de activación predeterminada suele ser “tanh”. No obstante, se pueden explorar otras funciones como “relu” o “sigmoid” según las necesidades del problema.
- **Función de Pérdida:** En problemas de regresión, es común utilizar el “mean squared error” (MSE) como función de pérdida. No obstante, la elección puede variar según el problema en particular.
- **Optimizador:** El “Adam” es un optimizador sólido y ampliamente utilizado en problemas de aprendizaje profundo. Aunque existen otras opciones como “RMSprop,” la elección depende en gran medida del contexto.
- **Tasa de Aprendizaje:** La tasa de aprendizaje es un hiperparámetro crucial. Por lo general, se comienza con un valor bajo, como 0.001, y se ajusta según la convergencia del modelo.
- **Dropout:** La técnica de dropout, aplicada en un porcentaje determinado de neuronas durante el entrenamiento, puede ayudar a prevenir el sobreajuste. Se recomienda iniciar con un valor del 20 % y ajustar según las necesidades del problema.

Es importante tener en cuenta que la elección de estos hiperparámetros puede depender en gran medida del problema específico y requiere experimentación para encontrar la configuración óptima.

Diseño y Entrenamiento del Modelo LSTM:

- Diseña un modelo LSTM.
- Compila el modelo con función de pérdida y optimizador.
- Entrena el modelo con datos de entrenamiento.

```
# Crear y compilar el modelo LSTM
model = Sequential()
model.add(LSTM(neuronas, activation=activacion, return_sequences=False,
               input_shape=(input_length, len(features))))
model.add(Dropout(0.3))
model.add(Dense(output_length, activation='linear'))
model.compile(optimizer=custom_RMS, loss='mean_squared_error')
```

Figura 3.15: Código de la estructura del LSTM.

Validación del Modelo:

Evalúa el rendimiento del modelo con datos de prueba.

Conjunto de prueba 1:

- División del conjunto de datos: 70 % train, 20 % validation, 10 % test
- Optimizador: Adam
- Pérdida: Mean Squared Error (mean_squared_error)
- Épocas: 200
- Tamaño de lotes: 30
- Estructura de red: 1 capa oculta con 64 neuronas
- Función de activación: Linear

```
# Crear el modelo LSTM
model = keras.Sequential([
    layers.LSTM(64, activation='linear', input_shape=(sequence_length, X_train.shape[1])),
    layers.Dense(1)
])

# Compilar el modelo
model.compile(optimizer='adam', loss='mean_squared_error')

# Entrenar el modelo
historial = model.fit(X_train_sequences, y_train_sequences, epochs=200, batch_size=30,
                      validation_data=(X_val_sequences, y_val_sequences), verbose=False)

# Evaluar el modelo en el conjunto de prueba
loss = model.evaluate(X_test_sequences, y_test_sequences)
val_loss = model.evaluate(X_val_sequences, y_val_sequences)
print("Pérdida en el conjunto de prueba:", loss)
print("Pérdida en el conjunto de prueba:", val_loss)
```

Figura 3.16: Código de la estructura del LSTM datos de prueba 1.

El código presentado (Figura 3.16) es un ejemplo de construcción, entrenamiento y evaluación de un modelo de red neuronal recurrente LSTM para un problema de regresión. Se inicia con la definición del modelo, que consta de una capa LSTM y una capa densa, seguido de la compilación con el optimizador Adam y la métrica de pérdida de error cuadrático medio. El modelo

se entrena durante 200 épocas con lotes de tamaño 30, utilizando datos de entrenamiento y validación. Luego se evalúa en un conjunto de prueba independiente y se calcula la pérdida en el conjunto de validación para supervisar el rendimiento del modelo. Las pérdidas resultantes se imprimen como indicadores de la calidad de las predicciones del modelo en diferentes conjuntos de datos.

```
1/1 [=====] - 0s 27ms/step - loss: 0.0182  
1/1 [=====] - 0s 28ms/step - loss: 0.0226  
Pérdida en el conjunto de prueba: 0.01822040230035782  
Pérdida en el conjunto de prueba: 0.022599518299102783
```

Figura 3.17: Rendimiento de datos de prueba 1.

27ms/step y 28ms/step: Estas cifras (Figura 3.17) indican el tiempo promedio que le toma al modelo procesar una iteración o paso de entrenamiento (epoch). En este caso, el modelo tarda aproximadamente 27-28 milisegundos en completar cada paso de entrenamiento. Este valor es útil para evaluar la eficiencia computacional del entrenamiento del modelo.

loss: 0.0182 y loss: 0.0226: Estos valores de la (Figura 3.17) representan la pérdida (loss) del modelo en los datos de entrenamiento en dos momentos diferentes del entrenamiento. La pérdida es una medida que indica cuán diferente son las predicciones del modelo de los valores reales en los datos de entrenamiento. En este contexto, una pérdida de 0.0182 y 0.0226 sugiere que el modelo está haciendo buenas predicciones en los datos de entrenamiento, con 0.0182 siendo un valor ligeramente mejor (menos pérdida) que 0.0226. Sin embargo, es importante recordar que estos valores solo representan el rendimiento del modelo en los datos de entrenamiento y no necesariamente reflejan su capacidad para generalizar en datos no vistos, como el conjunto de prueba.

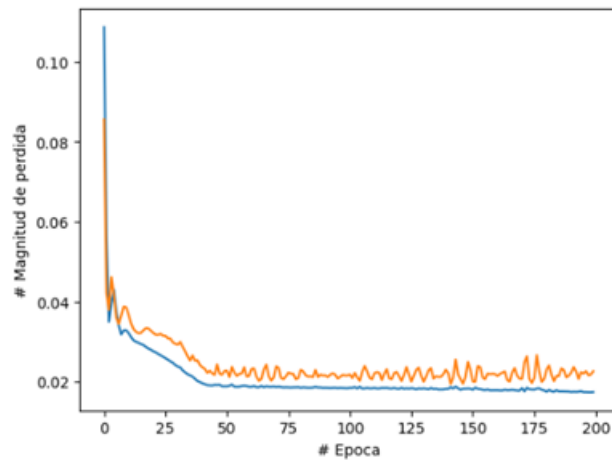


Figura 3.18: Grafico de predicción LSTM prueba 1

Conjunto de prueba 2:

- División del conjunto de datos: 70 % train, 15 % validation, 15 % test
- Optimizador: Adam (learning_rate=0.001)
- Pérdida: mean_squared_error
- Épocas: [50, 100, 200]
- Tamaño de lotes: [5, 10, 15]
- Estructura de red: 1 capa oculta con [32, 64, 128] neuronas
- Función de activación: relu

```
# Define una función que crea el modelo
def create_model(units=64):
    model = keras.Sequential([
        layers.LSTM(units, activation='relu', input_shape=(sequence_length, X_train.shape[1]), return_sequences=True),
        layers.Dense(1)
    ])
    model.compile(optimizer=keras.optimizers.Adam(learning_rate=0.001), loss='mean_squared_error')
    return model

# Configurar la validación cruzada con TimeSeriesSplit (especialmente útil para series temporales)
tscv = TimeSeriesSplit(n_splits=3)

best_score = float('inf')
best_params = {}

# Realizar la búsqueda de hiperparámetros
for units in [32, 64, 128]:
    model = create_model(units)
    for epochs in [50, 100, 200]:
        for batch_size in [5, 10, 15]:
            scores = []
            for train_index, val_index in tscv.split(X_train_sequences):
                X_train_split, X_val_split = X_train_sequences[train_index], X_train_sequences[val_index]
                y_train_split, y_val_split = y_train_sequences[train_index], y_train_sequences[val_index]
                model.fit(X_train_split, y_train_split, epochs=epochs, batch_size=batch_size, verbose=0)
                val_loss = model.evaluate(X_val_split, y_val_split)
                scores.append(val_loss)
            avg_score = np.mean(scores)
            if avg_score < best_score:
                best_score = avg_score
                best_params['units'] = units
                best_params['epochs'] = epochs
                best_params['batch_size'] = batch_size

# Entrenar el modelo con los mejores hiperparámetros encontrados
best_model = create_model(best_params['units'])
Historial = best_model.fit(X_train_sequences, y_train_sequences, epochs=best_params['epochs'], batch_size=best_params['batch_size'],
```

Figura 3.19: Código de la estructura del LSTM datos de prueba 2.

El código (Figura 3.19) presenta un proceso de búsqueda de hiperparámetros para un modelo LSTM en datos de series temporales. Se definen diferentes configuraciones de modelos con variaciones en el número de unidades LSTM, épocas de entrenamiento y tamaño de lote.

Estos modelos se evalúan en un esquema de validación cruzada específico para series temporales, y se promedian las puntuaciones de rendimiento en las divisiones de validación.

Los valores de hiperparámetros que generan la mejor puntuación se almacenan en `best_params`.

Luego, se crea un modelo con estos hiperparámetros óptimos y se entrena en todo el conjunto de datos de entrenamiento.

Esta metodología busca encontrar la configuración de modelo que se ajuste mejor a los datos de series temporales, garantizando que el modelo sea robusto y pueda generalizar efectivamente a datos futuros.

```
6/6 - 0s - loss: 0.0113 - val_loss: 0.0530 - 109ms/epoch - 18ms/step
1/1 [=====] - 0s 49ms/step - loss: 0.0266
1/1 [=====] - 0s 55ms/step - loss: 0.0530
Pérdida en el conjunto de prueba: 0.02664046362042427
```

Figura 3.20: Rendimiento de datos de prueba 2.

En el contexto de un modelo LSTM entrenado en series temporales, los valores proporcionados representan métricas clave para evaluar su desempeño (Figura 3.21). 6/6 indica que se han completado seis épocas de entrenamiento, mientras que 0s no especifica el tiempo transcurrido. La pérdida (loss) de 0.0113 en el conjunto de entrenamiento señala un buen ajuste del modelo a los datos de entrenamiento, mientras que la val_loss de 0.0530 en el conjunto de validación es ligeramente más alta, lo que sugiere que el modelo podría tener dificultades para generalizar. Los tiempos en milisegundos por época y por paso (109ms/epoch y 18ms/step) reflejan la eficiencia de entrenamiento del modelo. En resumen, estas métricas ofrecen información sobre la capacidad del modelo para aprender de los datos de entrenamiento y su habilidad para generalizar efectivamente a datos no vistos en el conjunto de validación, lo que es crucial en aplicaciones de series temporales.

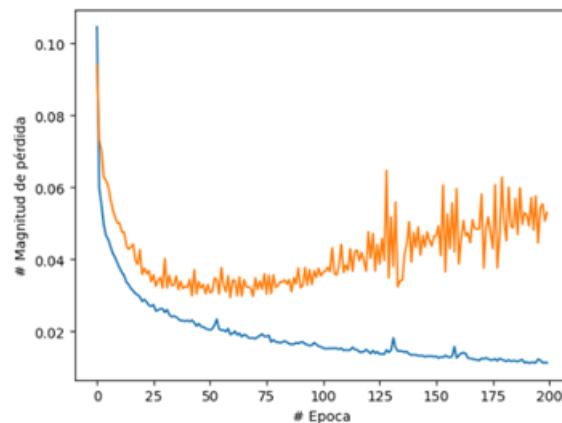


Figura 3.21: Grafico de predicción LSTM prueba 2

3.4. Prototipo del sistema

El sistema web se desarrollará utilizando una combinación de tecnologías, que incluye PHP para la lógica del servidor, JavaScript para la interactividad y dinamismo en el lado del cliente, y HTML y CSS para la estructura y el

diseño de la interfaz [45]. Esta elección de tecnologías permite la creación de una aplicación web versátil y receptiva, que ofrece una experiencia de usuario intuitiva y eficiente, al tiempo que garantiza un sólido procesamiento en el servidor para gestionar las diversas funcionalidades del sistema.

El nuevo modelo debe garantizar el abastecimiento oportuno de productos sin incurrir en sobre stock, minimizando el costo de inventario y facilitando el trabajo del área de compras. Para esto se propone diseñar y codificar una aplicación web que realice estas actividades, sin que los analistas deban realizar cálculos matemáticos complejos.

Levantamiento de requisitos

Importancia de la selección de una buena técnica de levantamiento de requerimientos Una técnica, es una serie de pasos documentados que van de la mano con unas reglas para su uso y criterios para verificar su corrección. Una técnica usualmente aplica a un proceso en el modelo de procesos. Algunas veces, dicha técnica incluye una notación y/o una herramienta asociada [46].

Requisitos del sistema

- Gestión de productos:
El sistema debe permitir la creación, modificación y eliminación de productos, incluyendo detalles como nombre.
- Órdenes de compra:
El sistema debe permitir la creación de órdenes de compra, especificando los productos y las cantidades.
- Órdenes de venta:
Debe ofrecer la capacidad de crear órdenes de venta para registrar las transacciones, especificando los productos, las cantidades.
- Generación de informe:
Capaz de generar informes sobre las predicciones de los productos.

En el contexto de esta tesis, las entrevistas se utilizaron como una técnica esencial para la recolección de datos y la comprensión de los requerimientos del sistema.

Las órdenes de compra son vitales para registrar las adquisiciones de insumos, ya que nos permiten especificar los productos y sus cantidades.

Asimismo, las órdenes de venta son esenciales para el registro de las ventas de productos finales, lo que proporciona un seguimiento preciso de las transacciones.

La generación de informes cobra importancia para analizar y predecir el comportamiento de los productos, permitiéndonos realizar ajustes necesarios en nuestras predicciones. En resumen, estos requisitos se originaron directamente de la necesidad de abordar eficazmente la problemática de la gestión de insumos gastronómicos y garantizar un funcionamiento óptimo del sistema.

El anexo de la tesis se convierte en un recurso valioso para aquellos lectores interesados en explorar en profundidad tanto la investigación cualitativa realizada a través de la entrevista como los aspectos técnicos y prácticos del prototipo desarrollado. Proporciona una visión completa de los dos componentes críticos de este trabajo: la obtención de requerimientos a través de entrevistas y la implementación técnica del sistema prototipo.

Capítulo 4

Resultados

4.1. Métricas de rendimiento

Las métricas de rendimiento son medidas cuantitativas utilizadas en aprendizaje automático para evaluar la calidad de los modelos predictivos. Estas métricas proporcionan una evaluación objetiva de la capacidad de un modelo para hacer predicciones precisas en tareas como clasificación o regresión.

4.1.1. Modelo de regresión lineal

los parámetros y configuraciones utilizados en el modelo de regresión con regularización Ridge:

- **loss=squared_loss:** Se ha elegido la función de pérdida de error cuadrático medio (MSE) debido a la naturaleza de un problema de regresión. La función MSE evalúa la discrepancia al cuadrado entre las predicciones y los valores reales, y se minimiza durante el proceso de entrenamiento.
- **penalty=l2:** La regularización Ridge ayuda a prevenir el sobreajuste incorporando un término de penalización en la función de pérdida.
- **alpha=0.0001:** En este caso, se ha optado por un valor pequeño (0.0001) para evitar una regularización excesiva y permitir que el modelo se ajuste de manera adecuada a los datos.
- **max_iter=1000:** Se ha establecido el número máximo de iteraciones permitidas para el descenso de gradiente en 1000, un valor suficientemente amplio para asegurar la convergencia del modelo.
- **tol=1e-3:** Este valor de tolerancia es apropiado para garantizar una convergencia razonable.

- **learning_rate=constant:** Se ha configurado la tasa de aprendizaje como constante a lo largo del proceso de entrenamiento.

```
# Crea el modelo de regresión lineal con SGD
model = SGDRegressor(
    loss="squared_error",
    penalty="l2", # Regularización Ridge
    alpha=0.0001, # Hiperparámetro de regularización
    max_iter=1000, # Número máximo de iteraciones
    tol=1e-3, # Tolerancia para la convergencia
    learning_rate="constant",
    eta0=0.01 # Tasa de aprendizaje inicial
)
model.fit(X_train, Y_train)

# Realizar predicciones en el conjunto de prueba
Y_train_pred = model.predict(X_train)
Y_train_pred = Y_train_pred.ravel()
Y_val_pred = model.predict(X_val)
Y_val_pred = Y_val_pred.ravel()
Y_test_pred = model.predict(X_test)
Y_test_pred = Y_test_pred.ravel()
```

Figura 4.1: Algoritmo de la estructura de regresión lineal.

A pesar de que tanto el Error Cuadrático Medio (MSE) como el Error Absoluto Medio (MAE) proporcionan indicadores favorables, el valor de R-cuadrado (R^2) indica que el modelo se encuentra notablemente alejado de la excelencia representada por un valor de 1, ya que $R^2 = 1$ constituye la condición ideal (Figura 4.2).

```
MSE en conjunto de validación: 0.05
MSE en conjunto de prueba: 0.03
MAE en conjunto de validación: 0.17
MAE en conjunto de prueba: 0.16
R^2 en conjunto de validación: 0.03
R^2 en conjunto de prueba: -0.02
```

Figura 4.2: Métrica de error regresión lineal.

4.1.2. Modelo LSTM

Al configurar una red neuronal LSTM, es común considerar varios hiperparámetros clave (Codigo 4.3):

- **Unidades de LSTM:** Se obtuvieron los mejores resultados con 64 neuronas.
- **Número de capas LSTM:** En este caso, se determinó que una sola capa oculta era la mejor opción.
- **Función de Activación:** Se lograron buenos resultados utilizando “relu” en la capa oculta y “linear” en la salida.
- **Función de Pérdida:** Se optó por el MSE debido a su versatilidad en problemas de regresión.
- **Optimizador:** En esta configuración, se prefirió “RMSprop” por su relevancia en predicciones de demanda.
- **Tasa de Aprendizaje:** Una tasa de aprendizaje de 0.01 resultó óptima.
- **Dropout:** Para esta configuración, un dropout del 20 % fue suficiente.

```
from keras.layers import Dropout
from keras.optimizers import Adam
from keras.optimizers import RMSprop

custom_RMS = RMSprop(learning_rate=0.01)
neuronas = 64
activacion = 'relu'
# Crear y compilar el modelo LSTM
model = Sequential()
model.add(LSTM(neuronas, activation=activacion, return_sequences=False, input_shape=(input_length, len(features))))
model.add(Dropout(0.2))
model.add(Dense(output_length, activation='linear'))
model.compile(optimizer=custom_RMS, loss='mean_squared_error')
```

Figura 4.3: Algoritmo de estructura LSTM.

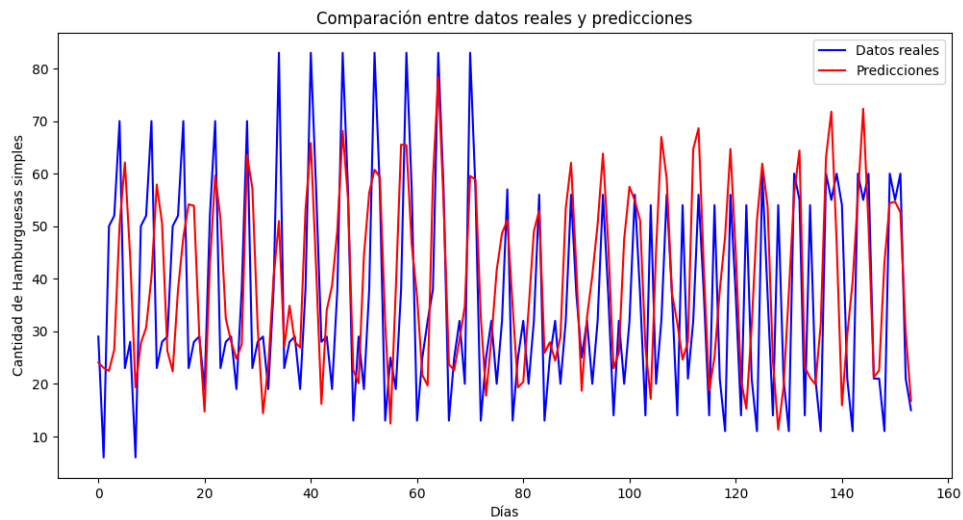


Figura 4.4: Grafico de predicción con LSTM.

Los tres valores muestran una proximidad aceptable, lo que indica que el modelo tiene una capacidad de generalización satisfactoria (Figura 4.5).

Comparativo desempeños:

RMS train: 0.020

RMS val: 0.039

RMS test: 0.026

Figura 4.5: Comparativo de desempeños LSTM.

En este trabajo para evaluar el rendimiento de una red neuronal LSTM se hace uso de la función de pérdidas (loss). Esta permite evaluar cómo es su evolución de los datos en un modelo. Según la literatura entre más grandes las pérdidas las predicciones van a estar alejadas del valor real. En la (Figura 4.6) se puede observar la evolución de pérdidas a través de las épocas, se evidencia dos señales que corresponden a entrenamiento y validación marcados con los colores azul y naranja respectivamente, se infiere que los valores decrecen con una alta tasa hasta la época 5 aproximadamente y luego no presentan cambios significativos. Las pérdidas son de 0.020 y 0.039 para entrenamiento y test respectivamente.

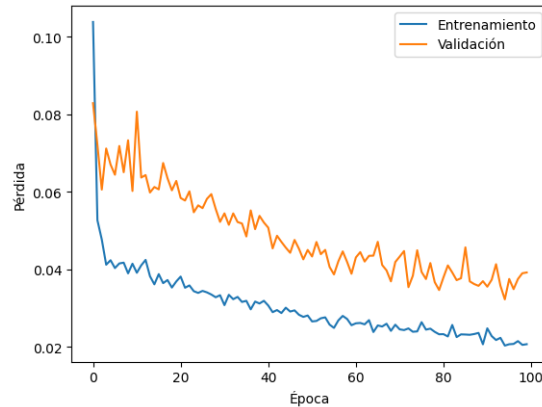


Figura 4.6: Grafico de perdida en las épocas LSTM.

4.2. Resultados de predicción de compras

Días	Real	Pred. Reg	Dif	Pred. LSTM	Dif
1	21	39	18	33	12
2	11	42	31	41	30
3	60	44	-16	52	-8
4	55	47	-8	47	-8
5	60	34	-26	36	-24
6	21	36	15	22	1
7	15	39	24	27	12
Dif. Total			38		15

Tabla 4.1: Tabla de resultados de las predicciones de ambos modelos

La tabla 4.1 proporciona un resumen de la predicción de la demanda de productos de la empresa gastronómica durante un período de siete días.

Compara los resultados entre los modelos de predicción estudiados: el modelo de regresión lineal y el modelo LSTM (Long Short-Term Memory).

En la columna “Días”, se enumeran los siete días del período en cuestión. En “Real”, se muestra la demanda real de productos para cada día, mientras que en “Pred. Reg” y “Pred. LSTM”, se presentan las predicciones realizadas por los modelos de regresión lineal y LSTM, respectivamente.

La columna “Dif” representa la diferencia entre las predicciones de cada modelo y la demanda real. En términos generales, se puede observar que ambas técnicas tuvieron algunas discrepancias con los valores reales. El modelo de regresión lineal tuvo una diferencia total de 38 unidades en comparación con la demanda real, mientras que el modelo LSTM tuvo una diferencia total de 15 unidades. En este caso, el modelo LSTM muestra un mejor desempeño al ser más preciso en la predicción de la demanda de productos en estos siete días en comparación con el modelo de regresión lineal.

En resumen, este estudio ha avanzado significativamente en la solución del problema de la predicción de la demanda semanal, y se espera que futuras investigaciones sigan ampliando los límites de la precisión y la aplicabilidad de estas predicciones en entornos de demanda.

Capítulo 5

Discusión

Se abordan las cuestiones de investigación planteadas, se muestra la consecución de los objetivos propuestos y, para concluir, se realiza un análisis con el propósito de validar las hipótesis formuladas en el contexto de este sistema de compra inteligente.

5.1. Logros alcanzados

- Se logró diseñar e implementar un prototipo funcional del sistema de compra inteligente. Este prototipo permite a los usuarios aprovechar el historial de ventas para tomar decisiones de compra más informadas y eficientes.
- Se realizó un exhaustivo análisis de datos para comprender los patrones de demanda y cómo estos pueden influir en las recomendaciones de compra. Esto incluyó la recopilación, limpieza y procesamiento de datos de demanda reales.
- Se desarrollaron y evaluaron algoritmos de recomendación personalizados que tienen en cuenta el historial de compras de los usuarios. Estos algoritmos ayudan a sugerir productos relevantes y optimizar las decisiones de compra.
- El trabajo aporta significativamente al campo de la inteligencia artificial y la toma de decisiones basada en datos al aplicar con éxito técnicas de predicción en el contexto de compras inteligentes.

5.2. Solución del problema de investigación

5.2.1. General

Desarrollar un sistema de compra inteligente basado en histórico de ventas para optimizar la gestión de compras de una empresa gastronómica de Ciudad del Este mediante algoritmos y técnicas de análisis de datos para predecir la demanda de productos en función del comportamiento de los clientes.

Al basarse en datos reales y en el comportamiento de los clientes, el sistema optimiza la gestión de compras al sugerir cantidades y variedades de productos a adquirir. De esta manera, se contribuye a una toma de decisiones más informada y eficiente en la adquisición de insumos gastronómicos, reduciendo costos y minimizando pérdidas por exceso o falta de inventario.

5.2.2. Específico

1. *Comprender en profundidad la lógica de negocio de la empresa gastronómica para identificar los procesos críticos que deben ser incluidos en el sistema informático a desarrollar.*

Se llevaron a cabo la investigación bibliográfica sobre el funcionamiento de empresas gastronómicas y visitas presenciales para observar directamente el funcionamiento de la empresa, anotando los procedimientos.

2. *Recabar los requisitos del sistema con los usuarios y partes interesadas.*

Para recopilar los requisitos del sistema con usuarios y partes interesadas, se utilizaron entrevistas detalladas. Esto permitió comprender sus necesidades y expectativas, y la información recopilada se documentó de manera organizada para definir el sistema con éxito.

3. *Modelar la lógica del sistema informático, utilizando técnicas y herramientas adecuadas, para garantizar su integridad y eficiencia.*

Se emplearon técnicas de modelado y herramientas especializadas para analizar datos históricos de demanda, patrones estacionales y tendencias. Esto permitió la creación de un modelo predictivo que mejora la eficiencia operativa al anticipar la demanda futura, optimizando la gestión de inventario y la asignación de recursos.

4. *Codificar el modelo lógico definido en lenguajes de programación Python y PHP.*

Se tradujo el modelo lógico previamente establecido en lenguajes de programación Python, PHP y otros, permitiendo la creación de algoritmos y funciones específicos para la predicción de la demanda. De esta manera, se garantiza que el sistema informático cuente con un componente crucial para la anticipación y gestión eficiente de la demanda, lo que contribuye a su integridad y eficiencia operativa.

5. *Depurar los datos cuantitativos recopilados sobre el comportamiento del consumidor.*

Se empleó la herramienta SQL para llevar a cabo tareas de limpieza y depuración de los conjuntos de datos, lo que incluyó la eliminación de errores, valores atípicos y datos inconsistentes. Esta depuración aseguró la integridad y calidad de los datos, lo que a su vez facilitó un análisis preciso y confiable. El resultado es una base de datos confiable y precisa que respalda la toma de decisiones informadas y estrategias efectivas.

6. *Realizar pruebas de usabilidad, accesibilidad, multiplataforma, y escalabilidad.*

Las pruebas se realizaron siguiendo un proceso estructurado. Para las pruebas de usabilidad, se involucraron usuarios reales o simulaciones, observando su interacción para evaluar la facilidad de uso. Las pruebas de accesibilidad se basaron en pautas de accesibilidad para garantizar el acceso para todos. En cuanto a las pruebas de multiplataforma, se verificó que el sistema funcionara correctamente en diversos dispositivos y navegadores. Las pruebas de escalabilidad implicaron pruebas de carga y rendimiento para evaluar la capacidad del sistema ante volúmenes crecientes de datos y demanda.

5.3. Análisis de hipótesis

El sistema de predicción cumple con la hipótesis de acierto en un 70 % no obstante no garantiza automáticamente una reducción del 10 % en el costo de materia prima. La relación entre la eficacia de la predicción y el ahorro de costos es influenciada por diversos factores, como la naturaleza específica del proceso de compra, la volatilidad del mercado, la precisión del modelo y la capacidad de adaptación del sistema a cambios en las condiciones del mercado.

5.4. Sugerencias para futuras investigaciones

Si bien este estudio alcanzó logros significativos, existen oportunidades para investigaciones futuras que ampliarían y mejorarían aún más la capacidad de pronóstico de la demanda. Las sugerencias para investigaciones futuras incluyen:

Exploración de técnicas avanzadas de aprendizaje profundo: La aplicación de técnicas de aprendizaje profundo más avanzadas, como redes neuronales recurrentes (RNN) bidireccionales, podría permitir la captura de patrones de demanda a largo plazo y mejorar aún más la precisión del pronóstico.

Consideración de datos externos: La inclusión de datos externos, como condiciones climáticas o promociones específicas, puede mejorar la capacidad de los modelos para adaptarse a eventos externos que afectan la demanda.

Automatización de búsqueda de hiperparámetros: La implementación de técnicas de optimización automática de hiperparámetros, como optimización bayesiana o búsqueda de cuadrícula inteligente, puede acelerar y mejorar el proceso de selección de hiperparámetros, lo que permitirá que los modelos se ajusten de manera más eficiente y logren una mayor precisión.

Desarrollo de una API de pronóstico: La creación de una API (Interfaz de Programación de Aplicaciones) permitiría a las empresas integrar fácilmente los modelos de pronóstico de demanda en sus sistemas existentes. Esto facilitaría la obtención de pronósticos en tiempo real y su uso en la toma de decisiones diarias de gestión de inventarios.

Anexos

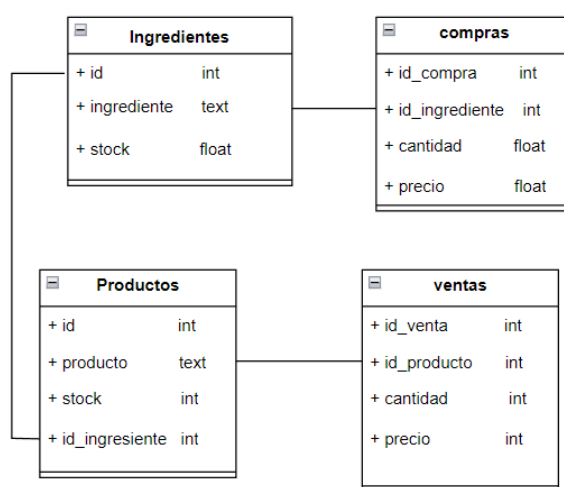
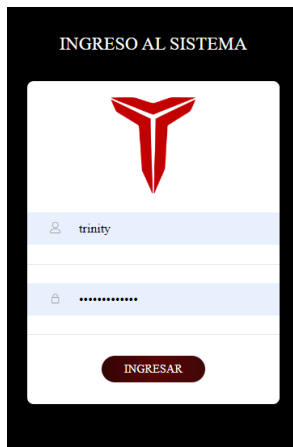


Figura 5.1: Diagrama de clases sistema de gestion de inventario.

El diagrama de clases del sistema comprende cuatro entidades principales: Compras, Ventas, Productos e Ingredientes. La clase Ingredientes incluye atributos como id, que funciona como clave primaria, el ingrediente en sí representado por la descripción y la cantidad en stock. Por otro lado, la clase Productos cuenta con atributos como id, que sirve como clave primaria, la descripción del producto, la cantidad en stock y una relación con la clase Ingredientes a través del id_ingrediente. De esta manera, las clases Compras y Ventas registran transacciones asociadas a estos productos e ingredientes, incluyendo detalles como el id_compra o id_venta, respectivamente, la cantidad y el precio. Este diseño de clases proporciona una estructura coherente para gestionar eficientemente las operaciones de compra y venta, manteniendo un control detallado sobre los productos e ingredientes disponibles en el sistema.



(a) Interfaz de inicio

```
<div class="limiter">
  <div class="container-login100" style="background-color:#000;">
    <div class="wrap-login100 p-t-30 p-b-50">
      <span class="login100-form-title p-b-41">
        Ingreso al sistema
      </span>
      <?php if (isset($_GET["error"])) { ?>
        <div class="alert alert-danger alert-dismissible fade in" role="alert">
          <button type="button" class="close" data-dismiss="alert" aria-label="Close">
            <span aria-hidden="true">x</span></button>
          <strong>Error!</strong> Usuario o contraseña incorrectos </div>
        <?php } ?>
        <form class="login100-form validate-form p-b-33 p-t-5" method="post">
          <br>
          <div align="center">
            
          </div>
          <div class="wrap-input100 validate-input" data-validate="Debe ingresar usuario">
            <input class="input100" type="text" autofocus name="user" placeholder="Usuario">
            <span class="focus-input100" data-placeholder="&#xe82a;"></span>
          </div>
          <div class="wrap-input100 validate-input" data-validate="Debe ingresar Contraseña">
            <input class="input100" type="password" name="pass" placeholder="Contraseña">
            <span class="focus-input100" data-placeholder="&#xe80f;"></span>
          </div>
          <div class="container-login100-form-btn m-t-32">
            <button class="login100-form-btn">
              Ingresar
            </button>
          </div>
        </form>
      </div>
    </div>
  </div>
</div>
```

(b) Código de inicio de sesión del sistema

Figura 5.2: Login del sistema

Este script PHP se utiliza para la autenticación de usuarios en un sistema web, empleando PHP y MySQL para gestionar la base de datos de usuarios. También se hace uso de tecnologías web como HTML, CSS. para el diseño del formulario de inicio de sesión. El script verifica las credenciales ingresadas por el usuario y, en caso de ser correctas, inicia una sesión y redirige al usuario a una página específica en función de su nivel de acceso. Si las credenciales son incorrectas, se muestra un mensaje de error.

X

Lista de Productos

ExcelPDFColumnas Visibles

Código	Marca	Categoría	Producto	Costo	Precio
1	POROTOS	Hamburguesas	Simple-porotito	15000	15000
2	POROTOS	Hamburguesas	Simple Bacon	20000	20000
3	POROTOS	Hamburguesas	Simple Cheddar	20000	20000
5	POROTOS	Hamburguesas	Simple S/V	15000	15000
6	POROTOS	Hamburguesas	Doble	20000	20000
7	POROTOS	Hamburguesas	Doble Bacon	25000	25000
8	POROTOS	Hamburguesas	Doble CHEDAR	25000	25000
10	POROTOS	Hamburguesas	Doble S/V	20000	20000
11	POROTOS	Hamburguesas	Triple	25000	25000
12	POROTOS	Hamburguesas	Triple Bacon	30000	30000

Buscar

Buscar

Buscar

Buscar

Buscar

Buscar

Mostrando página 1 de 5

Figura 5.3: Interfaz productos

X

Hamburguesa Simple

Ingredientes InicialesAgregados disponibles

Ingrediente

Cantidad

-- Seleccione el ingrediente --

0

Agregar

Estos ingredientes son los que vienen incluidos en el precio del producto.

Buscar:

Ingredientes Seleccionados	Cantidad	Acciones
Aro de cebollas	1 Gr	<div></div>
Carne	1 Gr	<div></div>
lechuga	1 Un	<div></div>
Pan blanco	1 Un	<div></div>
Queso Cheddar	1 Gr	<div></div>

Ingredientes Seleccionados

Cantidad

Acciones

Mostrando 1 a 5 de 6 registros

Anterior

1

2

Siguiente

Cancelar

Figura 5.4: Interfaz de carga de ingredientes

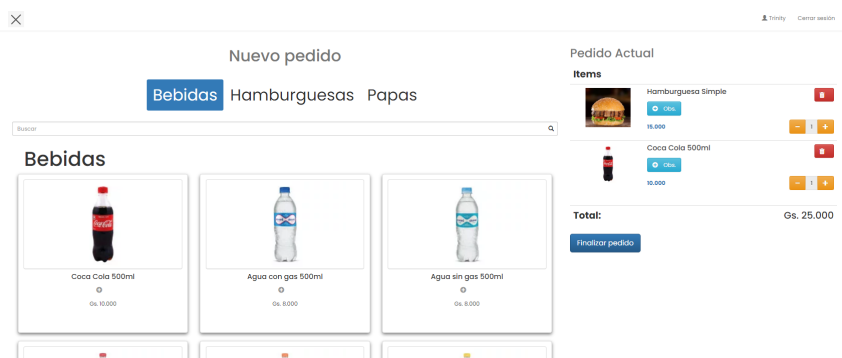


Figura 5.8: Interfaz de ventas

```
$(document).ready(function () {
    // $("#sidebar").toggleClass("active"); //ocultar sidebar al cargar
    // $("#sidebarCollapse").toggleClass("active"); //ocultar sidebar al cargar
});

function sumarCantidad(id, cantidad) {
    //peticion para sumar cantidad y recargar tabla
    setCantidad(id, (cantidad + 1));
}

function restarCantidad(id, cantidad) {
    if ((cantidad > 1)) return false;
    //peticion para sumar cantidad y recargar tabla
    setCantidad(id, (cantidad - 1));
}

function setCantidad(id, cantidad) {
    if (estaCargando()) return false; // si esta cargando, que no se haga la peticion

    $.ajax({
        method: "POST",
        url: "?c=presupuesto_tnp&a=SetCantidad",
        data: {
            id: id,
            cantidad: cantidad
        },
        success: function (respuesta) {
            $("#presupuesto_items").html(respuesta);

            Toast.fire({
                icon: 'success',
                title: 'Cantidad modificada'
            });
        },
        error: function (xhr, status) {
            Swal.fire({
                icon: 'error',
                title: 'Ocurrió un error',
                text: 'Revise su conexión a internet',
            })
        },
        complete: function (xhr, status) {
            // ocultar overlay
            removerAnimacionCarga();
        },
    });
});

function estaCargando() {
    //verificar si no esta cargando
    if ($("#overlay").hasClass('overlay')) {
        return true;
    }
    //coloca la clase overlay para evitar doble submit
    $("#overlay").addClass('overlay');
    $("#overlay").show();
    return false;
}

function removerAnimacionCarga() {
    $("#overlay").removeClass('overlay');
    $("#overlay").hide();
}

/* =====
busqueda en cards
===== */
$("#buscar-input").on("keyup", function () {
    var value = $(this).val().toLowerCase();
    // Si el valor es vacío, mostramos todos los elementos nuevamente
    if (value === '') {

```

Figura 5.9: Codigo carga de ventas

Lista de Predicciones

Nueva Predicción

Informe de compra semanal

Excel

PDF

Columnas Visibles

Fecha	Producto	Cantidades
05-10-2023 hasta 12-10-2023	Hamburguesa simple	[37,14,54,21,11,60,55]
29-09-2023 Hasta 05-10-2023	Hamburguesa simple	[57,13,25,32,20,32,56]
22-09-2023 Hasta 29-09-2023	Hamburguesa simple	[70,23,28,29,19,38,83]

Figura 5.10: Interfaz de lista de predicciones hechas

Lista de compras para la semana (predicción)

Producto	Cant	Precio compra	Total (Gs.)
Lechuga	50	2.000	100.000
Tomate	100	5.000	500.000
Pan Blanco	270	8.000	2.160.000
Carne	270	2.500	675.000
			Total Gs: 3.435.000

Figura 5.11: Informe de predicción de ingredientes a comprar

- 1- Se exporta el modelo en .h5

```
model.save('lstmdef.h5')
```

- 2- Se importa la librería en Python

```
!pip install tensorflowjs
```

- 3- Se crea una carpeta para el modelo

```
!mkdir salida
```

- 4- Se convierte el modelo a un modelo para tensorflow js

```
!tensorflowjs_converter --input_format keras lstmdef.h5 salida
```

- 5- Se generan dos archivos y los descargamos

group1-shard1of1.bin

model.json

Figura 5.12: Expotación de Python

- 1- Se copian los dos archivos "group1-shard1of1.bin" y "model.json" a la carpeta raíz del proyecto

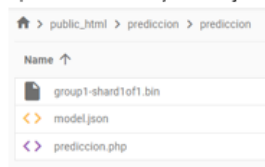


Figura 5.13: Importación a javascript

```
2- Se importa la librería tensorflow.js
<script src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs"></script>

3- Se hace la llamada al modelo
<script src="model.json"></script>

4- Se manda inputData
//Cargar modelo
(async () => {
    console.log("Cargando modelo...");
    modelo = await tf.loadLayersModel("model.json");
    console.log("Modelo cargado...");
    if (modelo != null) {
        // Convierte la matriz en un tensor
        const inputTensor = tf.tensor(inputData, shape, 'float32');

        var prediccion = modelo.predict(inputTensor).dataSync();

    } else {
        console.log("cargando modelo")
    }
})();

5- Se cargan los resultados a la base de datos
$.ajax({
    url: "cargar_prediccion.php",
    method: "POST",
    data: prediccion,
    cache: false,
    success: function (respuesta) {
        console.log("Cargado correctamente.");
    },
});
```

Figura 5.14: Importación a javascript

Referencias bibliográficas

- [1] Jimmy Alexander Romero Miranda, Víctor Alberto Lizcano Portilla *et al.*, “Predicción de ventas futuras,” 2021.
- [2] [En línea]. Disponible: <https://decidesoluciones.es/modelo-de-prediccion-de-demanda-aplicacion-y-beneficios/>
- [3] Ana Ortega Marqués, Sandy Patricia Padilla Domínguez, Johana Isabel Torres Durán, y Alexander Ruz Gómez, “Nivel de importancia del control interno de los inventarios dentro del marco conceptual de una empresa,” *Liderazgo estratégico*, vol. 7, no. 1, pp. 71–82, 2017.
- [4] Kenneth E Kendall y Julie E Kendall, *Análisis y diseño de sistemas*. Pearson educación, 2005, obtenido el 20 de mayo de 2023 de. [En línea]. Disponible: https://scholar.google.es/scholar?hl=es&as_sdt=0%2C5&q=K.+E.+Kendall+y+J.+E.+Kendall%2C+An%C2%B4alisis+y+dise%C2%B4on%2C+2005&btnG=
- [5] Xabier Basogain Olabe, “Redes neuronales artificiales y sus aplicaciones,” *Publicaciones de la Escuela de Ingenieros*, 1998.
- [6] César Pérez López y Daniel Santin González, *Minería de datos. Técnicas y herramientas: técnicas y herramientas*. Ediciones Paraninfo, SA, 2007.
- [7] Beatriz Beltrán Martínez, “Minería de datos,” *Cómo hallar una aguja en un pajar. Ingenierías*, vol. 14, no. 53, pp. 53–66, 2001.
- [8] Esteban Schab, Ramiro Rivera, Luciana Bracco, Facundo Coto, Patricia Cristaldo, Lautaro Ramos, Natalia Rapesta, Juan Pablo Núñez, Soledad Retamar, Carlos Casanova *et al.*, “Minería de datos y visualización de información,” in *XX Workshop de Investigadores en Ciencias de la Computación (WICC 2018, Universidad Nacional del Nordeste).*, 2018.

- [9] Alkis Simitsis y Panos Vassiliadis, “A method for the mapping of conceptual designs to logical blueprints for etl processes,” *Decision Support Systems*, vol. 45, no. 1, pp. 22–40, 2008, data Warehousing and OLAP. [En línea]. Disponible: <https://www.sciencedirect.com/science/article/pii/S0167923606002065>
- [10] Alexander Bustamante Martínez, Ernesto Amaru Galvis Lista, y Luis Carlos Gómez Flórez, “Técnicas de modelado de procesos de etl: una revisión de alternativas y su aplicación en un proyecto de desarrollo de una solución de bi,” *Scientia et technica*, vol. 18, no. 1, pp. 185–191, 2013.
- [11] Rodolfo H Villarroel, Yessica M Gómez, y Constanza B Krause, “Incorporación de seguridad en el modelado conceptual de procesos extracción-transformación-carga,” *Información tecnológica*, vol. 24, no. 6, pp. 101–110, 2013.
- [12] Dennys Fabian Herrera Cofre *et al.*, “Predicción para el mercado de acciones con redes neuronales lstm,” 2020.
- [13] Batta Mahesh, “Machine learning algorithms-a review,” *International Journal of Science and Research (IJSR)*. [Internet], vol. 9, no. 1, pp. 381–386, 2020. [En línea]. Disponible: https://scholar.google.es/scholar?hl=es&as_sdt=0%2C5&q=+machine+learning&btnG=
- [14] I.B.M. Staff(es). (2017) Machine learning. [En línea]. Disponible: <https://www.ibm.com/mx-es/analytics/machine-learning>
- [15] Carlos Arana, “Redes neuronales recurrentes: Análisis de los modelos especializados en datos secuenciales,” Serie Documentos de Trabajo, Tech. Rep., 2021.
- [16] Ian Goodfellow, Yoshua Bengio, y Aaron Courville, *Deep learning*. MIT press, 2016.
- [17] Vahid Mirjalili y Sebastian Raschka, *Python machine learning*. Marcombo, 2020.
- [18] E De la Rosa Montero, “El aprendizaje profundo para la identificación de sistemas no lineales,” *Centro de Investigación y de estudios avanzados del Instituto Politécnico nacional*, 2014.
- [19] Xavier Basogain, “Redes neuronales artificiales y sus aplicaciones,” *Dpto. Ingeniería de Sistemas y Automática, Escuela Superior*

- de Ingeniería Bilbao. Open Course Ware.*[En línea] disponible en http://ocw.ehu.es/enseñanzas-tecnicas/redes-neuronales-artificiales-y-sus-aplicaciones/Course_listing. [Consultada 20-09-2012], 2008.
- [20] Raquel Flórez López y José Miguel Fernández Fernández, *Las redes neuronales artificiales*. Netbiblo, 2008.
- [21] Nicolás Sánchez Anzola, “Máquinas de soporte vectorial y redes neuronales artificiales en la predicción del movimiento usd/cop spot intradiario.” *ODEON-Observatorio de Economía y Operaciones Numéricas*, no. 9, 2015.
- [22] Andres Nacelle y E Mizraji, “Redes neuronales artificiales,” *Núcleo de ingeniería biomédica-Universidad de la Republica Uruguay*, 2009.
- [23] Ian Goodfellow, Yoshua Bengio, y Aaron Courville, *Deep Learning*. MIT Press, 2016, <http://www.deeplearningbook.org>.
- [24] Javier Román Morales *et al.*, “Redes recurrentes para el análisis de series temporales,” B.S. thesis, 2018.
- [25] Besay Montesdeoca Santana, “Estudios de predicción en series temporales de datos meteorológicos utilizando redes neuronales recurrentes,” B.S. thesis, 2016.
- [26] Samuel Tomas, Oliver Saavedra, y Israel Espinoza, “Predicción del ciclo solar 25 mediante modelos arima y redes neuronales lstm,” *Revista de la Academia Colombiana de Ciencias Exactas, Físicas y Naturales*, 2023.
- [27] Juan José De Lucio Fernández, “Estimación adelantada del crecimiento regional mediante redes neuronales lstm,” *Investigaciones Regionales= Journal of Regional Research*, no. 49, pp. 45–64, 2021.
- [28] Carlos Andrés Sepúlveda Calle y Milton Tarsicio Benavides Posso, “Análisis predictivo de demanda de servicios bajo series temporales,” 2023.
- [29] María Carrasco Carrasco, “Técnicas de regularización en regresión: Implementación y aplicaciones,” 2016.
- [30] Eder Acevedo, Alexei Serna, y Edgar Serna, “Principios y características de las redes neuronales artificiales,” *Desarrollo e innovación en ingeniería*, vol. 173, 2017.

- [31] M Gestal Pose, “Introducción a las redes de neuronas artificiales,” *Departamento de Tecnologías de la Información y las Comunicaciones. Universidad da Coruña*, 2009.
- [32] Haward Miguel Chang Hidalgo, “Comparación de técnicas de estimación basadas en machine learning para predecir costos en los planes de adquisiciones de las entidades públicas del Perú,” 2023.
- [33] Pedro Santander, Claudio Elórtogui, Cristian González, Héctor Allende-Cid, y Wenceslao Palma, “Redes sociales, inteligencia computacional y predicción electoral: el caso de las primarias presidenciales de Chile 2017,” *Cuadernos. info*, no. 41, pp. 41–56, 2017.
- [34] Nicole Pacheco Allende, “Rediseño del proceso de abastecimiento en el área de compras en dos empresas dedicadas al equipamiento gastronómico,” 2015.
- [35] Víctor Marcial Condorena Rondón, “Desarrollo de un sistema de control de inventario, para la gestión de compras de materia prima en el rubro de restaurantes,” 2017.
- [36] Edwar Andrés Hincapié Herrera, “Predicción de la demanda usando modelos de machine learning,” 2021.
- [37] María Afanador Jiménez, Silvia Juliana Casadiegos Chaparro, Isabella Campo Maichel, Juan Sebastián Casallas Estrella *et al.*, “Diseño de un modelo de pronóstico de demanda basado en machine learning y un modelo multi-objetivo para planeación de la producción en una industria panificadora,” 2022.
- [38] Paul DuBois, *MySQL*. Addison-Wesley, 2013.
- [39] Álvaro Chilet Vera, “Elaboración de un algoritmo predictivo para la reposición de hipoclorito en los depósitos mediante técnicas de machine learning,” Ph.D. dissertation, Universitat Politècnica de València, 2023.
- [40] Numpy. [En línea]. Disponible: <https://numpy.org/>
- [41] The Pandas Development Team. Pandas documentation. [En línea]. Disponible: <https://pandas.pydata.org/about/index.html>
- [42] Matplotlib Development Team. (2023) Matplotlib official website. [En línea]. Disponible: <https://matplotlib.org/>

- [43] Google Brain Team. Tensorflow. [En línea]. Disponible: <https://www.tensorflow.org/>
- [44] Keras. (2023) Keras documentation. <https://keras.io/>. Accessed: 2023-08-10.
- [45] Robin Nixon, *Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5*. .o'Reilly Media, Inc.", 2014.
- [46] Nicolás Aristizábal Mejía y Miguel Eduardo Torres Moreno, "Técnicas de levantamiento de requerimientos con innovación," in *Presentado en el Cuarto Congreso Colombiano de Computación 4CCC. Sociedad Colombiana de Computación S (CO)*, vol. 2, 2009.
- [47] Jorge Mohedano, José Miguel Saiz, y Pedro Salazar Román, *Iniciación a javascript*. Ministerio de Educación, 2012.