UNIVERSIDAD POLITÉCNICA ESTATAL DEL CARCHI

FACULTAD DE COMERCIO INTERNACIONAL, INTEGRACIÓN, ADMINISTRACIÓN Y ECONOMÍA EMPRESARIAL

CARRERA DE LOGÍSTICA Y TRANSPORTE

Tema: "Diseño de un programa para la gestión de inventarios basado en árbol de decisiones que permita la optimización de recursos en las empresas"

Trabajo de Integración Curricular previo a la obtención del título de Ingenieros en Logística y Transporte

AUTORES: Ruano Piscal Luis Stiven

Tapia Collaguazo Henry Stalin

TUTOR: Msc. Iván Gabriel Mafla Bolaños

Tulcán, 2025.

CERTIFICADO DEL TUTOR

Certifico que los estudiantes Ruano Piscal Luis Stiven y Tapia Collaguazo Henry Stalin con el número de cédula 0450071592 y 1724754724 respectivamente han desarrollado el Trabajo de Integración Curricular: "Diseño de un programa para la gestión de inventarios basado en árbol de decisiones que permita la optimización de recursos en las empresas"

Este trabajo se sujeta a las normas y metodología dispuesta en el Reglamento de la Unidad de Integración Curricular, Titulación e Incorporación de la UPEC, por lo tanto, autorizo la presentación de la sustentación para la calificación respectiva

IVAN GABRIEL MAFLA BOLAÑOS

Firmado digitalmente por IVAN GABRIEL MAFLA BOLAÑOS
Fecha: 2025.03.24 17:10:16 -05'00'

Msc. Iván Gabriel Mafla Bolaños

TUTOR

Tulcán, marzo de 2025

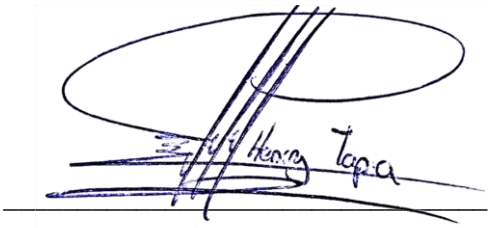
AUTORÍA DE TRABAJO

El presente Trabajo de Integración Curricular constituye un requisito previo para la obtención del título de Ingenieros en la Carrera de logística y transporte de la Facultad de Comercio Internacional, Integración, Administración y Economía Empresarial

Nosotros, Ruano Piscal Luis Stiven y Tapia Collaguazo Henry Stalin con cédula de identidad número 0450071592 y 1724754724 respectivamente declaramos que la investigación es absolutamente original, auténtica, personal y los resultados y conclusiones a los que hemos llegado son de nuestra absoluta responsabilidad.

Ruano Piscal Luis Stiven

AUTOR



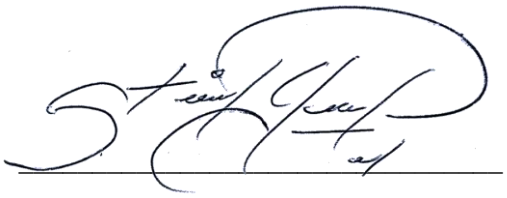
Tapia Collaguazo Henry Stalin

AUTOR

Tulcán, marzo de 2025

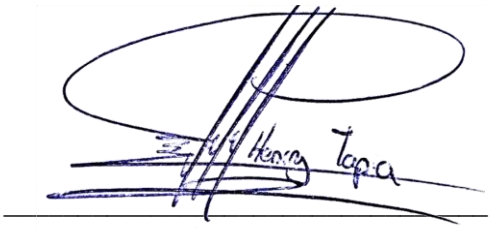
ACTA DE CESIÓN DE DERECHOS DEL TRABAJO DE INTEGRACIÓN CURRICULAR

Nosotros Ruano Piscal Luis Stiven y Tapia Collaguazo Henry Stalin declaramos ser autores de los criterios emitidos en el Trabajo de Integración Curricular: "Diseño de un programa para la gestión de inventarios basado en árbol de decisiones que permita la optimización de recursos en las empresas" y se exime expresamente a la Universidad Politécnica Estatal del Carchi y a sus representantes de posibles reclamos o acciones legales.



Ruano Piscal Luis Stiven

AUTOR



Tapia Collaguazo Henry Stalin

AUTOR

Tulcán, marzo de 2025

AGRADECIMIENTO

Quiero expresar mi más profundo agradecimiento a mi familia por su apoyo incondicional a lo largo de esta etapa universitaria, especialmente a mi madre, quien estuvo a mi lado en cada momento, brindándome su amor y fortaleza.

Agradezco también a mis profesores de la carrera de Logística y Transporte, quienes, al compartir su conocimiento y experiencia, han dejado una huella significativa en mi formación profesional, conocimientos que estoy seguro serán valiosos en mi futuro laboral. Extiendo un especial reconocimiento a mi tutor, Msc. Iván Mafla, por su paciencia, compromiso y dedicación, factores clave para la culminación exitosa de mi trabajo de titulación.

Finalmente, quiero expresar mi agradecimiento a todos los amigos que formaron parte de mi camino universitario, en especial a Stiven Ruano, cuyo apoyo fue invaluable tanto en el desarrollo de mi trabajo de titulación como en nuestra amistad. Gracias a ellos, esta etapa no solo resultó enriquecedora, sino también una de las más memorables y significativas de mi vida. **Henry Tapia**

AGRADECIMIENTO

Al concluir este trabajo, deseo recordar cada etapa que he superado en mi vida y expresar mi gratitud a todas las personas que, con su apoyo incondicional, han sido fundamentales en la realización de este proyecto de investigación. A mi familia, cuyo ejemplo de esfuerzo, disciplina y constancia me ha inspirado diariamente a seguir mis sueños. Su lucha incansable y su amor incondicional han sido el motor que me ha impulsado a enfrentar cada desafío y alcanzar mis metas. A mis amigos de clase, con quienes compartí momentos inolvidables durante esta etapa de formación y en especial a Henry Tapia, por su apoyo constante y la amistad inquebrantable que ha marcado cada paso de mi carrera.

A la Universidad Politécnica Estatal de Carchi, a la Carrera de Logística y Transporte, y a mis docentes, quienes no solo compartieron su conocimiento, sino también su dedicación, paciencia y motivación, dejando una huella imborrable en mi formación. Quiero expresar mi más sincero agradecimiento al MSc. Iván Mafla por su valioso apoyo, confianza y orientación durante el desarrollo de este proyecto de tesis. Sus consejos y el tiempo que me brindó han sido invaluables para alcanzar este logro.

Stiven Ruano

DEDICATORIA

Dedico este trabajo de titulación y todos los logros que alcance en mi vida a mi madre, la persona más importante para mí, cuya presencia y amor incondicional me han guiado siempre. También lo dedico a mi padre, quien, aunque no pudo estar para ver este logro, fue una parte fundamental en mi camino hasta aquí. Su recuerdo y enseñanza siguen siendo una fuente constante de inspiración. **Henry Tapia**

DEDICATORIA

Primeramente, quiero agradecer Luis Ruano y Gloria Piscal, quienes han sido el mayor ejemplo de amor, esfuerzo, perseverancia y superación en mi vida. Agradezco por inculcarme valores que han forjado mi carácter, moldeado mi pensamiento y enriquecido mis virtudes. Por estar a mi lado como guardianes incansables de mis sueños y ser la luz que orienta mis pasos hacia un futuro lleno de esperanza.

A mis hermanos Byron, Lorena y Ángelo, les agradezco por ser mi impulso en los momentos más difíciles. Gracias por brindarme su apoyo incondicional, por alentarme cuando todo parecía desmoronarse y por confiar siempre en mí, incluso cuando yo dudaba. Su amor ha sido mi refugio y su mano, la fuerza que me levantó cuando más lo necesitaba.

A mis cuñados William, Elizabeth y Dianele por sus valiosos consejos y su inquebrantable apoyo, que me motivaron a seguir adelante y alcanzar mis sueños.

A mis sobrinos Bright, Matías, Zhair, Ariel, Luna, Haylen y Jakob, les agradezco por su apoyo incondicional y por siempre creer en mí. Espero que este logro sirva como un ejemplo de que, a pesar de las dificultades, la perseverancia, la dedicación y el amor familiar nos permiten alcanzar cualquier objetivo.

Finalmente, quiero dedicar este logro a mi niño interior, ese que en los inicios experimentó miedo y dudas. A ese niño que soñaba, pero también temía, le agradezco por no rendirse. Y esa diminuta voz constante y silenciosa que me estuvo presente en los momentos más difíciles, recordándome la importancia de seguir adelante. **Stiven Ruano**

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

ÍNDICE DE ANEXOS

RESUMEN

Esta investigación desarrolló un programa de gestión de inventarios en *Python* utilizando árboles de decisión y redes neuronales, basándose en modelos determinísticos y probabilísticos. El programa calcula la cantidad óptima de pedido y el punto de reorden, entre otros valores relevantes según el tipo de inventario. Para el análisis de variables, se aplicaron técnicas de validación cruzada con árboles de decisión y redes neuronales, combinando métodos de predicción y clasificación. En el árbol de decisiones de predicción, se utilizó *MAE* = mediana ($|\hat{y}i - yi|$) para evaluar la predicción, obteniendo un valor óptimo de *MAE*= 2,99, mientras que en clasificación alcanzó una precisión del 90% de predicciones correctas. Posteriormente, se implementaron redes neuronales para mejorar la capacidad predictiva, usando el MSE = $\frac{1}{n}\sum_{i=1}(Y_i - \hat{Y}_i)^2$ en predicción, alcanzando un valor óptimo de 0,902, y la entropía cruzada Loss $=-\frac{1}{N}\sum_{i=1}^{N}(y_i \log(p_i) + (1-y_i)\log(1-p_i)$ en clasificación, logrando un valor óptimo de 0,614, y un valor óptimo de precisión del 69%. Para finalizar realizando una comparativa de los programas usando *MSE* y *RMSE* = $\sqrt{\frac{1}{n}\sum_{i=1}(Y_i - \hat{Y}_i)^2}$, evidenciando que las redes neuronales aportaron una mayor precisión en las predicciones que los árboles de decisión.

Palabras Claves: modelos de inventario, Árbol de decisión, Red neuronal, *Python*, probabilístico, determinístico

ABSTRACT

This research developed an inventory management program in Python using decision trees and neural networks, based on deterministic and probabilistic models. The program calculates the optimal order quantity and reorder point, among other relevant values depending on the inventory type. For variable analysis, cross-validation techniques were applied with decision trees and neural networks, combining prediction and classification methods. In the predictive decision tree, MAE = median ($|\hat{y}i - yi|$) was used to evaluate predictions, achieving an optimal MAE value of 2.99, while classification reached 90% accuracy in correct predictions. Subsequently, neural networks were implemented to enhance predictive capacity, using MSE = = $\frac{1}{n}\sum_{i=1}(Y_i - \hat{Y}_i)^2$ or prediction, achieving an optimal value of 0.902, and cross-entropy Loss = $-\frac{1}{N}\sum_{i=1}^{N}(y_i\, log(p_i) + (1 - y_i) log\ (1 - p_i)$ for classification, obtaining an optimal value of 0.614 and an accuracy rate of 69%. Finally, a comparative analysis of the programs using MSE and RMSE = $\sqrt{\frac{1}{n}\sum_{i=1}(Y_i - \hat{Y}_i)^2}$, demonstrated that neural networks provided greater prediction accuracy than decision trees.

Keywords: *Inventory Models, Decision Tree, Neural Network, Python, Probabilistic, Deterministic*

INTRODUCCIÓN

En la obra de Guerrero (2022), se indica que una gestión eficiente de inventarios no solo permite reducir los costos vinculados al almacenamiento de productos, sino que también incrementa la productividad operativa, generando una ventaja competitiva en mercados donde la demanda es altamente variable. Las estrategias de control de inventarios facilitan que las empresas reaccionen rápidamente ante los cambios en el mercado. No obstante, según Cruz (2017) en países en desarrollo, la adopción de tecnologías de gestión de inventario puede resultar complicada debido a limitaciones económicas y a una infraestructura tecnológica deficiente, lo que impide el avance hacia la modernización y dificulta la capacidad de reaccionar rápidamente a los cambios del mercado.

En Ecuador, un gran número de empresas aún utiliza métodos tradicionales, como el uso de tarjetas *Kardex*, lo que genera ineficiencias operativas, incrementa los costos y propicia errores humanos (Vásconez et al, 2020). Estos factores obstaculizan la toma de decisiones rápidas y acertadas. En contraste, las grandes corporaciones a nivel internacional han adoptado tecnologías avanzadas que optimizan la cadena de suministro y mejoran la gestión de inventarios.

Este estudio tiene como objetivo diseñar un programa de gestión de inventarios basado en un árbol de decisiones, que brinde a las empresas una solución tecnológica accesible y eficiente. El programa permitirá optimizar recursos, reducir costos y mejorar la toma de decisiones en la cadena de suministro. Entre sus funcionalidades, el sistema calculará la cantidad económica de pedido y el punto de reorden, además de incorporar múltiples modelos de inventarios, tanto determinísticos como probabilísticos, adaptándose a las diversas necesidades de las empresas, sin importar su tamaño o sector.

La relevancia de esta investigación radica en que muchas empresas de la región carecen de los conocimientos o los recursos necesarios para implementar sistemas de inventario más sofisticados, lo que repercute en su competitividad, así lo expone Cruz (2017). El desarrollo de este programa, utilizando el lenguaje de programación *Python*, ofrecerá una herramienta robusta y fácil de usar, que no requerirá grandes inversiones en infraestructura o capacitación especializada. Se espera que, al adoptar esta tecnología, las empresas no solo mejoren su eficiencia operativa, sino

que también fortalezcan su capacidad para responder de manera ágil a las demandas del mercado, contribuyendo así al crecimiento de la economía local.

Por tal motivo, esta investigación busca contribuir a la optimización de los recursos empresariales, mejorando la eficiencia de los procesos, reduciendo errores operativos y aumentando la rentabilidad a largo plazo, lo que permitirá a las empresas competir de manera más efectiva en el mercado.

I. EL PROBLEMA

1.1. PLANTEAMIENTO DEL PROBLEMA

En la obra de Vásquez (2020), se observa cómo las grandes empresas a nivel internacional manejan enormes cantidades de inventarios y ocupan grandes espacios de almacenamiento donde se complica llevar un control eficiente en todas estas actividades. Si estas actividades se realizan solo con recursos humanos pueden llegar a ser contraproducentes para la empresa, ya que para el personal llevar el registro de todas las existencias que tiene la empresa, tanto lo que se recibe como lo que se despacha, llega a ser demasiado trabajo si se manejan enormes cantidades de inventario y varios despachos al día. Por eso, las grandes empresas han implementado herramientas tecnológicas para aumentar su productividad teniendo un buen manejo de inventarios, lo que les da una ventaja competitiva.

En su obra Carvajal, Burgos y Hermida (2017) señalan que, en Ecuador existe un pequeño porcentaje de empresas que, al igual que las empresas internacionales, ya tienen sistematizadas gran parte de sus actividades. Utilizando herramientas tecnológicas para el mejor desarrollo de sus actividades y administración de sus recursos, logrando tener gran ventaja sobre el resto de las empresas del país. Sin embargo, Ecuador aún es un país en vías de desarrollo en términos tecnológicos. Las empresas del país en su gran mayoría tienen pocos recursos y conocimientos en temas de sistematización de procesos; la mayoría de sus actividades no cuentan con un sistema de optimización tecnológica en sus procesos. En tema de inventarios, muchas son las empresas que siguen usando las tarjetas *Kardex* como sistema de administración. En pleno 2023, son muchas las opciones para reemplazar este sistema ya muy obsoleto y antiguo.

Puetate (2014) señala que en la ciudad de Tulcán existen algunas empresas que ya cuentan con procesos sistematizados, pero son muy escasas. La principal razón de esto es la falta de conocimiento que tienen las empresas para adoptar estos sistemas, las capacitaciones necesarias para manejarlos y el miedo a adaptarse a nuevos campos tecnológicos. Estas empresas necesitan con mucha urgencia un sistema que

les permita manejar sus inventarios de mejor manera para evitar problemas que pueden afectar toda la cadena de suministros. Son problemas que se resolverían fácilmente con una sistematización simple del manejo de inventarios que no requeriría de un gran esfuerzo y capital para integrarlo a sus operaciones. Una vez implementado un sistema tecnológico a su sistema de inventarios, la toma de decisiones se realizaría de una forma más rápida, precisa y eficiente.

Este estudio tiene como objetivo desarrollar un sistema tecnológico basado en árboles de decisión para optimizar la gestión de inventarios y la toma de decisiones en esta área. El sistema permitirá evaluar diversos resultados, incluyendo cantidades de pedido según un artículo, varios artículos, según un periodo, varios periodos, con restricción de almacén, con costo de preparación, sin costo de preparación y con discontinuidad de precios. Así como también el punto de reorden para cada uno de estos casos.

1.2. FORMULACIÓN DEL PROBLEMA

¿Cómo se pueden optimizar los recursos de las empresas mediante el diseño de un sistema de inventarios basado en árbol de decisiones?

1.3. JUSTIFICACIÓN

La presente investigación tiene como objetivo desarrollar e implementar un sistema de gestión tecnológica que servirá para optimizar y potenciar la capacidad de toma de decisiones en el área de inventarios. Sin embargo, cabe resaltar que este sistema no va a realizar acciones tan básicas como controlar entradas y salidas, si no que se va a enfocar en algo más complejo, como es calcular la cantidad de pedido y el punto de reorden para cada modelo de inventario que existe, se tomara un enfoque diferente ya que la gran mayoría de sistemas de gestión de inventarios calcula entradas y salidas, así que se optó por realizar un programa diferente.

Para dar respuesta al problema planteado, se realizó un diagnóstico del estado actual de la gestión de inventarios sin la intervención de ningún sistema tecnológico. Se identificaron las problemáticas asociadas a la falta de dicho sistema, así como los beneficios de implementar un diseño tecnológico en la gestión de inventarios.

Además, se consideraron los retos inherentes a esta implementación, como la necesidad de adquirir conocimientos y habilidades para manejar estos sistemas. Aunque estos conocimientos son básicos, existe un gran porcentaje de personas que no dominan adecuadamente el uso de una computadora.

Para la programación del software, se optó por utilizar lenguajes de programación que permitan desarrollar un software amigable visualmente y fácil de utilizar, al mismo tiempo que sea robusto. Por esta razón, se buscará el lenguaje de programación más adecuado y que cumpla con los requisitos necesarios para crear esta aplicación.

Al implementar este software para el diseño de un sistema de gestión de inventarios, se evitarán errores humanos y se mejorará la eficiencia operativa, además de potenciar la capacidad de toma de decisiones.

Esta investigación tiene como objetivo proporcionar apoyo a las empresas ubicadas en la ciudad de Tulcán. Se centra específicamente en aquellas empresas debidamente registradas en la Cámara de Comercio de Tulcán. Dentro de este grupo, el enfoque se dirige exclusivamente hacia las empresas de comercialización que realizan actividades mínimas de abastecimiento, almacenamiento y ventas. El objetivo es colaborar con estas empresas para optimizar la gestión de sus sistemas de inventario, promoviendo la adopción de tecnologías, especialmente en los procesos logísticos. Al aprovechar los beneficios que la tecnología puede ofrecer, se aspira a impulsar mejoras continuas y abrir oportunidades para aumentar la competitividad a nivel nacional. Este enfoque no solo fortalecerá la confianza de los clientes, sino que también mejorará la experiencia que estas empresas ofrecen.

La teoría de sistemas estudia los diferentes sistemas y subsistemas que tiene una un organismo que en este caso son las partes o sistemas que hacen que una empresa funciones. Con relación a la cadena de suministros esta teoría pretende identificar las bases teórico - metodológicas que influyen en la formación de la cadena y los tipos de relación entre sus elementos (Tapia, 2014).

Se eligió esta teoría como base, ya que puede utilizarse para optimizar el diseño de un sistema tecnológico de gestión de inventarios. Esta decisión facilitará la definición clara del sistema, estableciendo sus límites, entradas, salidas y componentes. A continuación, se llevará a cabo un análisis exhaustivo de la interacción entre los subsistemas de recepción, almacenamiento, preparación de pedidos y distribución.

El objetivo es mejorar la rentabilidad, la rotación y la productividad, al mismo tiempo que se minimizan los desabastecimientos, las pérdidas y las rupturas de *stock*.

Otra teoría importante para tomar en cuenta en este proyecto es la teoría de la complejidad computacional. Según Arora y Barak (2009) la teoría de la complejidad computacional, una rama de la teoría de la computación se enfoca en la clasificación de problemas computacionales según su dificultad mediante el uso de algoritmos. Este campo de estudio se dedica a analizar los recursos necesarios para abordar problemas, teniendo en cuenta aspectos como el tiempo y el espacio requeridos. En el programa en cuestión, que implica trabajar con algoritmos de diversas complejidades, esta teoría se considera fundamental. Resultará relevante al seleccionar y establecer los algoritmos más eficientes para resolver los problemas que puedan surgir durante el desarrollo del programa.

1.4. OBJETIVOS Y PREGUNTAS DE INVESTIGACIÓN

1.4.1. Objetivo General

Diseñar un programa para la gestión de inventarios basado en árbol de decisiones que permita la optimización de recursos en las empresas

1.4.2. Objetivos Específicos

1. Describir el estado actual de todos los procesos de inventarios que realiza una empresa sin ningún tipo de sistema tecnológico
2. Analizar todos los modelos de inventarios existentes según las características propias de cada empresa y de la demanda.
3. Determinar el lenguaje más adecuado para la programación del sistema de gestión de inventarios.
4. Diseñar y desarrollar un programa para la gestión de inventarios
5. Realizar una simulación de los procesos de gestión de inventarios con el programa ya implementado

1.4.3. Preguntas de Investigación

1. ¿Cuál es el estado actual de los procesos de inventarios que realiza una empresa sin ningún tipo de sistema tecnológico?
2. ¿Cuáles son los modelos de inventarios existentes según las características propias de cada empresa y de la demanda?
3. ¿Cuál el lenguaje más adecuado para la programación del sistema de gestión de inventarios?
4. ¿Cómo diseñar y desarrollar un programa para la gestión de inventarios?
5. ¿Cómo se simula los procesos de inventarios con la aplicación ya implementada a los procesos?

II. FUNDAMENTACIÓN TEÓRICA

2.1. ANTECEDENTES DE LA INVESTIGACIÓN

Aguilar (2021) desarrolló un modelo de basado en tecnología para mejorar la eficiencia en los procesos de pedidos a proveedores de TUS ENKNTOS, un negocio mayorista de productos de perfumería, cosméticos y artículos de aseo personal en Tulcán. Para diseñar este modelo, se realizó primero una encuesta semiestructurada para recopilar información sobre el estado actual de los procesos de pedidos. Se concluyó que el modelo propuesto optimiza estos procesos, ya que permite al negocio determinar la cantidad adecuada de pedidos.

Chávez (2021) se basó en un análisis de datos secundarios, que contribuyeron a evaluar la situación logística actual de la empresa. A través de la aplicación de modelos de inventario, se logró optimizar los recursos de la organización. Se integraron al estudio herramientas cuantitativas como el cálculo del pronóstico para el año 2020. Se llegó a la conclusión de que, con los diferentes cálculos aplicados, se obtuvo un resultado del 63% de optimización del inventario, lo cual indica un eficaz manejo de los recursos logísticos identificados en la cadena logística de la empresa.

Valera (2023) desarrolló una investigación orientada a mejorar la gestión logística en el sector eléctrico del cono norte de Lima durante el año 2022, con el objetivo de optimizar los recursos de los fabricantes de ferretería. El estudio empleó un enfoque cuantitativo y un método hipotético-deductivo, permitiendo concluir que diversas actividades y estrategias pueden medir la efectividad de los controles internos a través de los resultados obtenidos. Asimismo, se realizaron recomendaciones para mejorar los procesos logísticos y gestionar de manera más eficiente los recursos empresariales."

2.2. MARCO TEÓRICO

2.2.1. Árbol de decisiones

Esta técnica, desarrollada por Howard Raiffa, se fundamenta en diferentes tipos de nudos y sus resultados esperados. Estos nudos incluyen los de decisión, que se representan con un cuadrado y marcan momentos en los que es necesario tomar una decisión en situaciones de incertidumbre; los nudos de sucesos, simbolizados por un círculo, que indican puntos de incertidumbre en términos de probabilidad; y los resultados esperados que dependen tanto de cada decisión como de los sucesos que ocurren (Córdoba, 2014).

En los árboles de decisiones, se combinan nudos de decisión y de eventos, y las decisiones deben tomarse considerando la incertidumbre. Siempre se comienza con un nudo de decisión, a partir del cual se despliegan todas las posibles consecuencias a lo largo del tiempo que puede resultar de cada elección.

2.2.2. Procedimiento hacia atrás

Según Córdoba (2014) :

> "Este procedimiento implica resolver primero las decisiones más alejadas del origen, y a partir de los resultados obtenidos, continuar resolviendo las decisiones inmediatamente previas hasta llegar a la primera decisión, la que está más a la izquierda" (pág. 310).

2.2.3. Optimización de recursos

Cortina (2016) señala que la optimización de recursos se relaciona con la eficiencia, entendida como el uso óptimo de los recursos para obtener mayores beneficios al menor costo posible.

2.2.4. Almacén

Tejero (2008) señala que el almacén es el "lugar donde físicamente se almacenan los *stocks* de productos comerciales o industriales que posteriormente serán objeto de distribución o transformación" (pág. 19).

2.2.5. Almacenamiento

Serrano (2019) subraya la importancia de "colocar la mercancía en la ubicación más adecuada del almacén para facilitar su acceso y localización" (pág. 18). Este principio tiene como objetivo optimizar el flujo de trabajo en el almacén, ya que una

organización adecuada de los productos ayuda a reducir los tiempos de búsqueda y manipulación, mejorar la eficiencia operativa y minimizar los costos relacionados con el movimiento de mercancías. La disposición estratégica de los artículos también favorece una gestión más ágil y ordenada del inventario, garantizando una respuesta rápida a las necesidades de los clientes y a los procesos productivos.

2.2.6. Gestión y Control de existencias

En la obra de Serrano (2019), se discute la relevancia de establecer la cantidad óptima de almacenamiento para cada producto, así como de calcular la frecuencia y el volumen de los pedidos. Este enfoque tiene como finalidad reducir los costos de almacenamiento al evitar el exceso de inventario y disminuir el capital inmovilizado. Al optimizar estos aspectos, se obtienen beneficios como una mayor eficiencia en la gestión de inventarios y una disminución de los costos asociados al mantenimiento del *stock*, lo que a su vez mejora la rentabilidad y asegura la disponibilidad de productos en el momento adecuado.

2.2.7. Inventarios

Meana (2017) describe el proceso como la verificación y control de los recursos o bienes patrimoniales de la empresa, llevados a cabo para regularizar las cuentas de existencias contables y determinar si se han producido pérdidas o ganancias. Esta definición resalta la importancia de realizar una supervisión continua y rigurosa de los inventarios, con el fin de asegurar que los registros contables sean precisos y reflejen correctamente la situación real de las existencias. Este proceso es esencial para identificar discrepancias, prevenir pérdidas y optimizar la toma de decisiones tanto financieras como operativas.

2.2.8. Manejo o control de inventarios

El inventario implica la confirmación y verificación de todos los tipos de existencias disponibles en la empresa, lo cual se realiza a través de un recuento físico de los productos. Los inventarios proporcionan información actualizada sobre el valor de las mercancías que poseemos (Meana, 2017).

2.2.9. *Stock*

La gestión de inventarios se define como "una acumulación de materiales y/o productos finales almacenados para su posterior venta al cliente; una inversión en *stock* que retiene recursos económicos durante un período" (Meana, 2017, pág. 4).

Esta definición resalta que mantener inventarios conlleva un costo relacionado con el capital inmovilizado, lo que puede afectar la liquidez de la empresa. Para abordar este problema, es crucial garantizar que el *stock* tenga una rotación efectiva, lo que implica que los productos se vendan o utilicen de manera eficiente y oportuna, reduciendo así el tiempo en que los recursos están retenidos y minimizando el riesgo de obsolescencia o deterioro.

2.2.10. Existencias

Meana (2017) indica que, en el campo de la gestión de inventarios, "las existencias son los productos que la empresa posee en sus instalaciones para ser vendidos al cliente final, así como aquellos que serán requeridos en algún momento durante su proceso de producción" (pág. 4). Este concepto subraya la importancia de mantener un nivel adecuado de inventario para garantizar la continuidad del proceso productivo y satisfacer eficientemente la demanda del cliente, evitando interrupciones y optimizando los recursos de la empresa.

2.2.11. *Stock* mínimo o *Stock* de seguridad

Carro y Gonzáles (2013) sostienen que "las empresas mantienen un inventario de seguridad. Este inventario actúa como una protección contra la incertidumbre en la demanda, los plazos de entrega y el suministro" (pág. 4). Esta afirmación destaca la relevancia de contar con un inventario de seguridad, que permite a las empresas reducir los riesgos asociados con la cadena de suministro, tales como las variaciones en la demanda, las demoras en las entregas y la inestabilidad en el abastecimiento. Así, el inventario de seguridad ayuda a asegurar la continuidad de las operaciones y a satisfacer las expectativas de los clientes.

2.2.12. Punto de reorden

Castillo (2014) destaca que el punto de reorden es un concepto esencial en la gestión de inventarios. Se define como el momento en el que "se estima la demanda, lo que permite determinar la cantidad necesaria para el reabastecimiento en el próximo período, así como el instante en que se debe realizar un nuevo pedido basado en una cantidad fija" (pág. 7). Este enfoque es crucial para mantener niveles óptimos de inventario, ya que facilita la planificación oportuna del reabastecimiento, minimizando los riesgos de desabastecimiento y asegurando la continuidad del proceso productivo y la satisfacción de la demanda del cliente.

2.2.13. Costos de inventarios

2.2.13.1. Costos de mantenimiento

Meana (2017) señala que el almacenamiento de productos conlleva una serie de costos significativos que deben tenerse en cuenta en la gestión de inventarios. Estos gastos incluyen el alquiler del espacio de almacenamiento, servicios básicos como agua y electricidad, limpieza, así como los costos relacionados con el deterioro y la caducidad de los productos. Además, mantener un alto nivel de *stock* genera costos adicionales, como los asociados al capital inmovilizado y el riesgo de obsolescencia. Todos estos aspectos subrayan la importancia de una gestión eficiente del inventario para reducir costos y optimizar los recursos de la empresa.

2.2.13.2. Costos de pedido

Castillo (2005) señala que los costos de transporte de un pedido comprenden todos los gastos relacionados con el traslado y la gestión de los artículos adquiridos. Estos costos incluyen actividades como la compra, la preparación de especificaciones y documentos, la emisión de órdenes de compra, el seguimiento a los proveedores y la inspección de los pedidos al momento de su recepción. Este conjunto de actividades es fundamental para asegurar que los pedidos lleguen en las condiciones esperadas y a tiempo, optimizando así el proceso logístico y manteniendo la eficiencia en la cadena de suministro.

2.2.14. Modelos de Inventarios

2.2.14.1. Determinísticos

En la obra de Meana (2017) se presenta un modelo en el que se considera que la demanda se mantiene constante a lo largo del tiempo. Este enfoque parte de la premisa de que las necesidades de los clientes son estables, lo que facilita una planificación más precisa de los inventarios. Al no existir variaciones en la demanda, es posible determinar el tiempo óptimo para el reabastecimiento y gestionar las existencias de manera más eficiente, garantizando así que las necesidades del cliente se satisfagan de forma continua y confiable.

2.2.14.2. Probabilísticos

Cuando la demanda es impredecible y no se puede estimar con exactitud, es fundamental tener un *stock* de seguridad para prevenir desabastecimientos. En este escenario, se realiza un pedido cuando las existencias actuales están a punto de

terminar, lo que ayuda a asegurar la disponibilidad de productos y a minimizar el riesgo de quedarse sin inventario (Meana, 2017).

2.2.15. Lenguaje de programación *Python*

Python es una herramienta para la elaboración de programas que adopta un enfoque orientado a objetos, así como también imperativo y funcional, siendo un lenguaje multiparadigma que se basa en el lenguaje ABC. Incluye estructuras de datos como listas, diccionarios, conjuntos y tuplas, las cuales permiten al usuario realizar tareas complejas en pocas líneas de código y de manera legible (Díaz, García, y Challenge, 2014).

2.2.16. Redes neuronales

Las redes neuronales permiten extraer información de manera eficaz y generar inferencias a partir de los datos disponibles, gracias a su capacidad de aprendizaje. Se caracterizan por su habilidad para identificar patrones mediante el entrenamiento con datos. Además, son sistemas autoajustables, lo que les permite aprender y adaptarse a través del ajuste de las conexiones en cada una de las neuronas que componen el sistema (Manguay y Vallejo, 2022).

2.2.17. Cantidad a pedir

La cantidad a pedir, también conocida como *EOQ* (*Economic Order Quantity*), es fundamental en la gestión de inventarios. Se define como "la cantidad óptima de inventario que se debe ordenar para minimizar los costos totales de inventario, incluyendo los costos de mantenimiento y de pedido" (Heizer y Render, 2014, pág. 490). Este modelo ayuda a las empresas a equilibrar los costos asociados con el mantenimiento de inventarios y los costos de realizar nuevos pedidos.

2.2.18. Visual Studio Code

Este editor se ejecuta en el escritorio y es compatible con *Windows*, *macOS* y *Linux*. Además, ofrece soporte integrado para lenguajes como *JavaScript*, *TypeScript* y *Node.js*, y cuenta con un amplio ecosistema de extensiones que permite la integración de otros lenguajes y entornos de desarrollo. Esta capacidad de adaptación lo convierte en una opción ideal para trabajar en diversas plataformas y con múltiples lenguajes de programación (Microsoft, 2021).

2.2.19. Desviación estándar

En el ámbito del análisis estadístico, la desviación estándar juega un papel crucial. Se la define como una medida de dispersión para variables cuantitativas y se calcula como la raíz cuadrada de la varianza (Calderón, 2015).

2.2.20. Error cuadrático medio

Es una medida utilizada para evaluar la calidad de un estimador. Se define como la esperanza del cuadrado de la diferencia entre el estimador $\hat{\theta}$ y el parámetro verdadero θ (Benjamín, 2014).

$$\boldsymbol{ECM_0[\hat{\theta}] = E_0\left[(\hat{\theta} - \theta)^2\right]} \quad (1)$$

2.2.21. Error absoluto

Es la diferencia entre el valor real y el valor medido de una determinada magnitud.

$$\boldsymbol{EA = Valor\ medio - Valor\ real} \quad (2)$$

El valor real es generalmente desconocido, por lo que se estima que el error absoluto es igual a la menor división de la escala del instrumento de medición (Lupo y Rigalli, 2024).

III. METODOLOGÍA

3.1. ENFOQUE METODOLÓGICO

3.1.1. Enfoque

3.1.1.1. Mixto

"Los métodos mixtos representan un conjunto de procesos sistemáticos, empíricos y críticos de investigación e implican la recolección y el análisis de datos cuantitativos y cualitativos, así como su integración y discusión conjunta, para realizar inferencias producto de toda la información recabada (meta inferencias) y lograr un mayor entendimiento del fenómeno bajo estudio" (Hernandez, Carlos, y Lucio, 2014, pág. 534).

El desarrollo de la investigación se basará en obtener una comprensión completa de la problemática, combinando datos cuantitativos con percepciones cualitativas sobre la resistencia al cambio y la necesidad de desarrollo tecnológico. Esto proporcionará una base sólida para la toma de decisiones y la implementación de soluciones efectivas en la gestión de inventarios.

3.1.2. Tipo de Investigación

3.1.2.1. Investigación Descriptiva

"La investigación descriptiva busca especificar propiedades, características y rasgos importantes de cualquier fenómeno que se analice" (Hernandez, Carlos, y Lucio, 2014, pág. 92)

Para llevar a cabo esta investigación, se empleó un enfoque de investigación que es apropiado debido a que permite describir los procesos de gestión de pedidos de manera efectiva.

3.1.2.2. Investigación – acción

Yuni y Urbano (2005) mencionan que "Se enmarca en un modelo de investigación de mayor compromiso con los cambios sociales, por cuanto se fundamenta en una

posición respecto al valor intrínseco que posee el conocimiento de la propia práctica y de las maneras personales de interpretar la realidad para que los propios actores puedan comprometerse en procesos de cambio personal y organizacional" (págs. 138 - 139)

Se utilizó este enfoque con el propósito de que la investigación no se limite únicamente a la observación y el diagnóstico, sino que también incluya la acción concreta de desarrollar y aplicar soluciones prácticas para resolver los problemas identificados en el ámbito de la gestión de inventarios.

3.2. HIPÓTESIS

H0 = Un programa para la gestión de inventarios basado en árbol de decisiones no permite la optimización de recursos en las empresas.

H1 = Un programa para la gestión de inventarios basado en árbol de decisiones permite la optimización de recursos en las empresas.

3.3. DEFINICIÓN Y OPERACIONALIZACIÓN DE LAS VARIABLES

Tabla 1. Operacionalización de las variables

Variables	Definición	Dimensiones	Indicadores	Técnicas	Instrumento
Independiente: Sistema de gestión de inventarios	Los sistemas de gestión de inventarios son procedimientos basados en modelos determinísticos y probabilísticos de cálculo de las cantidades óptimas a solicitar de cada uno de los ítems almacenados (Bustos & Chacón, 2007).	Compras	Costo de compras Tiempo de Ciclo de Compra Proceso de compras Cantidad mensual de pedido	Entrevista Estructurada	Guía de preguntas
		Almacenamiento	Costo de almacenamiento Proceso de almacenamiento Asignación de espacios		
		Ventas	Tiempo Promedio de Ciclo de Ventas Promedio de clientes Comportamiento demanda		
		Modelos	Parámetros de cada inventario Tipo de sistema de revisión Tipo de demanda	Análisis documental	Ficha
Dependiente:			Precisión de Inventarios	Entrevista	

Variables	Definición	Dimensiones	Indicadores	Técnicas	Instrumento
Optimización de recursos	La optimización de recursos se refiere a la forma de mejorar alguna acción o trabajo realizada, esto nos da a entender que la optimización de recursos es buscar la forma de mejorar el recurso de una empresa para que esta tenga mejores resultados, mayor eficiencia o mejor eficacia. (Herrera, 2017)	Uso Eficiente de Inventarios	Uso de tecnologías Cantidad económica de pedido Punto de reorden	Estructurada Análisis documental	Guía de preguntas Ficha
		Programación	Cumplimiento de Plazos Retrasos en la planificación Cumplimiento de pedidos Eficiencia en la gestión de pedidos Lenguajes de programación	Entrevista Estructurada Análisis documental	Guía de preguntas Ficha

3.4. MÉTODOS UTILIZADOS

3.4.1. Metodología Deductiva

Se empleará este enfoque de investigación, basándose en conocimientos previamente adquiridos en programación, gestión de inventarios, almacenamiento y árbol de decisiones, para aplicarlos en la creación y desarrollo del programa de gestión de inventarios.

3.4.2. Metodología Inductiva

Para la investigación, también sería prudente utilizar la metodología inductiva. Se analizarán casos específicos para determinar la eficiencia comparativa de diversos modelos de inventarios. Este análisis proporcionará certeza sobre las fortalezas específicas de cada modelo de inventario. Con esta información, se desarrollará un programa que abarque una gestión más general de todos los inventarios, no limitándose a modelos específicos.

3.4.3. Técnicas

3.4.3.1. Análisis documental

Al revisar la documentación adecuada, esta puede proporcionar un registro histórico detallado de las transacciones de inventario, las diversas formas de gestionarlo y cómo estructurarlo. Esto permite comprender patrones de demanda, ciclos de inventario y tendencias estacionales, información clave para el desarrollo del programa.

3.4.3.2. Documentos, registros, materiales y artefactos

Esta técnica ofrece una fuente de datos objetiva y tangible que permite una comprensión completa de la situación actual del inventario y conocimientos más profundos de programación. Estos elementos proporcionan información precisa sobre inventarios y técnicas de programación. Al tener datos reales y verificables, se pueden tomar decisiones basadas en hechos concretos, optimizar los niveles de inventario, minimizar pérdidas y evitar errores humanos. Toda esta información y recursos se utilizarán para el desarrollo del programa.

3.5. ANÁLISIS ESTADÍSTICO

La presente investigación se enfoca en el análisis estadístico relacionado con las redes neuronales durante la ejecución de los códigos correspondientes. A lo largo del desarrollo de este trabajo, se ha podido observar que los análisis estadísticos desempeñan un papel crucial en la optimización y mejora de los modelos de redes neuronales. Estos análisis no solo han proporcionado una comprensión más profunda de los datos utilizados, sino que también han sido fundamentales para ajustar los parámetros y validar los resultados obtenidos. En consecuencia, el empleo de técnicas estadísticas ha facilitado de manera significativa el proceso de desarrollo, asegurando así la efectividad y precisión del modelo final.

En la sección de código dedicada a la evaluación del modelo, se encontró el error cuadrático medio, el cual se utiliza para realizar una matriz, evaluando la precisión de los modelos. Esto se logra mediante la comparación de la diferencia promedio entre los valores predichos por el modelo y los resultados reales. También es importante tener en cuenta que se penalizan los errores grandes, ya que se eleva al cuadrado las diferencias. Por lo tanto, estos valores pueden ser sensibles.

$$\boldsymbol{MSE} = \frac{\mathbf{1}}{\boldsymbol{n}} \sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} (\boldsymbol{y_i} - \boldsymbol{\hat{y}_i})^{\mathbf{2}} \tag{3}$$

De igual manera, en la sección del código dedicada a evaluar el modelo, se encontró que el error absoluto es fundamental para calcular el promedio de los errores en las predicciones del modelo. Este método se basa en tomar el valor absoluto de la diferencia entre las predicciones y los valores reales.

$$\boldsymbol{MAE} = \frac{\mathbf{1}}{\boldsymbol{n}} \sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} |\boldsymbol{y_i} - \boldsymbol{\hat{y}_i}| \tag{4}$$

Por otro lado, en la sección de código destinada a escalar los datos, se encontró que la diferencia de medianas es útil, ya que se basa en una medida de tendencia central entre dos conjuntos de datos, a diferencia de las medias, que pueden verse

afectadas por valores atípicos. La principal ventaja de la diferencia de medianas es su robustez; el valor de la mediana divide un conjunto de datos en dos partes iguales.

$$\boldsymbol{Diferencia\ de\ Medianas\ = Mediana\ (A) - Mediana\ (B)} \quad (5)$$

Esto ayudó a lograr un análisis más preciso en la obtención de los resultados. Estos métodos permiten realizar comparativas de los comportamientos observados y obtener resultados más exactos. A su vez, ayudan a verificar si los resultados obtenidos concuerdan con las precisiones esperadas.

IV. RESULTADOS Y DISCUSIÓN

4.1. RESULTADOS

La sección de resultados presenta el proceso completo, las actividades y la información requerida para alcanzar los diversos objetivos de este trabajo, junto con las conclusiones y resultados obtenidos. Esta sección culmina con la creación y prueba del programa desarrollado.

4.1.1. Describir el estado actual de todos los procesos de inventarios que realiza una empresa sin ningún tipo de sistema tecnológico

Para alcanzar este objetivo, se aplicaron dos técnicas de recolección de datos: entrevistas estructuradas y análisis documental. Estas técnicas fueron implementadas en las empresas registradas en la Cámara de Comercio de Tulcán, que se dedican principalmente a actividades de abastecimiento, almacenamiento y ventas. La Cámara proporcionó un listado de empresas, a partir del cual se llevó a cabo un sondeo para identificar aquellas que cumplían con los criterios establecidos. Aunque no todas respondieron positivamente a la solicitud de participación, se logró la colaboración de 10 empresas.

A pesar de que el número de empresas no alcanzó las expectativas, se obtuvo la cooperación de una variedad de sectores, que incluyen prendas de vestir, repuestos de bicicletas y motocicletas, electrodomésticos, tecnología, víveres, e industria de vidrio y aluminio. Esta diversidad en la información recolectada permitirá adaptar el programa a diferentes tipos de empresas, cumpliendo así con los objetivos de la investigación.

En relación con este objetivo, se examinaron las entrevistas realizadas, centrándose en los procesos de compras, almacenamiento y ventas, así como en el uso de tecnología en dichas actividades. Se observó que muchas empresas empleaban medios tecnológicos en algunos de estos procesos, aunque no todos los aspectos

estaban cubiertos por la tecnología, sino únicamente uno o dos específicos. Por lo tanto, se realizó una clasificación de los procesos según su uso de tecnología.

Para la realización de las tablas 2 a la 6, se obtuvo la información recopilada del Anexo 3.

Para un análisis más detallado de la información recopilada, primero fue necesario identificar los procesos que estas empresas gestionan mediante tecnología. En la Tabla 2 se presentan estos procesos.

Tabla 2. Actividades que hacen uso de algún sistema tecnológico

Uso de sistema tecnológico			
Empresa	**Compras**	**Almacenamiento**	**Ventas**
A TODP PEDAL	No se utiliza	No se utiliza	No se utiliza
Abastos el Nazareno	No se utiliza	No se utiliza	No se utiliza
Águila Bike	No se utiliza	Se utiliza	Se utiliza
Comercial Enríquez	No se utiliza	No se utiliza	No se utiliza
Ferroalvid	No se utiliza	Se utiliza	Se utiliza
Mega Ofertas	No se utiliza	No se utiliza	No se utiliza
Movitech	No se utiliza	Se utiliza	No se utiliza
SALOMON	No se utiliza	No se utiliza	No se utiliza
Technet	No se utiliza	No se utiliza	No se utiliza
Tejidos Olarte	No se utiliza	No se utiliza	No se utiliza

En la Tabla 2, se detallan las empresas entrevistadas y se indica qué procesos emplean algún sistema tecnológico y cuáles no. A partir de esta información, se enfocarán en los procesos que no hacen uso de tecnología, ya que son indispensables para el análisis y la creación de un proceso general para las empresas que operan sin sistemas tecnológicos.

Una vez identificados los procesos que se llevan a cabo sin tecnología, es fundamental también determinar el tiempo requerido para cada una de estas actividades. En la Tabla 3 se presentan los de la gestión de compras.

Tabla 3. Tiempo promedio de compras

Tiempo de proceso de compras (minutos)	
Empresa	**Compras**
A TODP PEDAL	20
Abastos el Nazareno	15
Águila Bike	25
Comercial Enríquez	25
Ferroalvid	40
Mega Ofertas	30
Movitech	40
SALOMON	90
Technet	60
Tejidos Olarte	30
Tiempo Promedio	37,5

Al confeccionar la Tabla 3 con los tiempos promedio, lo ideal sería calcular el número de observaciones necesarias. Basándose en los tiempos promedio obtenidos, hubiera sido necesario realizar aproximadamente 30 observaciones para cada empresa en este proceso específico. Sin embargo, esto habría implicado esperar a cada empresa en alrededor de 30 ocasiones para observar y medir los tiempos de compra. Según las entrevistas realizadas, las compras se efectúan una vez al inicio de cada mes o cada 15 días, lo que habría requerido una cantidad excesiva de tiempo para recopilar esta información.

Por esta razón, se optó por realizar diversas entrevistas a las empresas. No obstante, estas entrevistas no fueron llevadas a cabo con empleados sin un amplio conocimiento de la empresa, sino que se dirigieron a expertos en esta materia. Por lo tanto, los tiempos fueron extraídos basándose en la experiencia que poseen estos expertos en dichas actividades. Este objetivo fue desarrollado a través de entrevistas con expertos, con el objetivo de evitar un proceso de observación que habría sido demasiado extenso.

En las Tablas 4 a 6 se muestran los tiempos promedio calculados para cada actividad analizada en cada empresa.

Tabla 4. Tiempo promedio de almacenamiento

Tiempo de proceso de almacenamiento (minutos)	
Empresa	**Almacenamiento**
A TODP PEDAL	180
Abastos el Nazareno	60
Comercial Enríquez	40
Mega Ofertas	20
SALOMON	150
Technet	40
Tejidos Olarte	60
Tiempo Promedio	78,5

Tabla 5. Tiempo promedio de registro de productos

Tiempo de proceso de registro de productos (minutos)	
Empresa	**Registro**
A TODP PEDAL	60
Abastos el Nazareno	30
Comercial Enríquez	35
Mega Ofertas	20
SALOMON	60
Technet	15
Tejidos Olarte	60
Tiempo Promedio	40

Tabla 6. *Tiempo promedio de ventas*

Tiempo de proceso de ventas (minutos)	
Empresa	**Ventas**
A TODP PEDAL	15
Abastos el Nazareno	20
Comercial Enríquez	10
Mega Ofertas	30
Movitech	30
SALOMON	10
Technet	30
Tejidos Olarte	30
Tiempo Promedio	21,8

Este análisis proporcionó los tiempos promedio de los procesos logísticos realizados por empresas que no utilizan ningún sistema tecnológico. Ahora, con estos datos en mano, se complementará la información con los detalles obtenidos en las entrevistas, especialmente sobre los procedimientos de compras, almacenamiento y ventas. Al fusionar esta información con los tiempos promedio, se elaborará un flujograma del proceso general de todas las actividades que una empresa lleva a cabo sin recurrir a tecnología alguna.

Para confeccionar el siguiente flujograma, se concentró los esfuerzos en las interrogantes clave de las entrevistas, especialmente en lo referente a los procedimientos de compras, ventas y almacenamiento. Además, se incorporó el tiempo requerido para el registro de entradas y salidas, un paso adicional esencial para gestionar inventarios de manera eficiente. Posteriormente, de entre las respuestas recopiladas, se seleccionaron únicamente aquellos procesos que todas las empresas realizan de manera consistente, independientemente del tipo de producto que ofrecen como se muestra en la Figura 1.

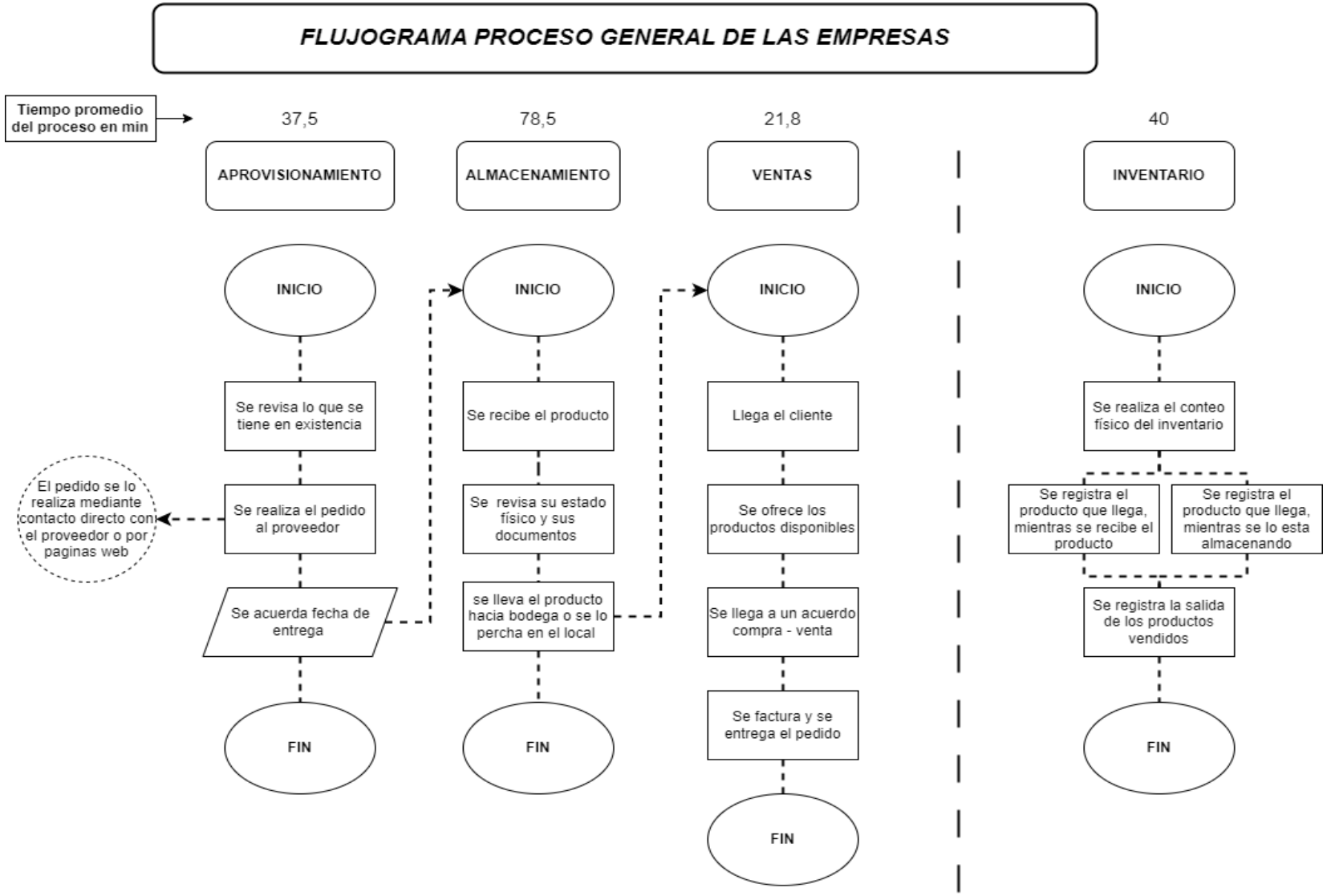


Figura 1. Flujograma proceso general de las empresas

4.1.2. Analizar todos los modelos de inventarios existentes según las características propias de cada empresa y de la demanda.

El resultado de este objetivo fue analizar en profundidad las características y el funcionamiento de las diferentes empresas con el fin de diseñar el programa de gestión de inventarios y adaptarlo de mejor manera a las distintas empresas analizadas. Para ello, es fundamental comprender a cabalidad cómo operan y se desenvuelven estas organizaciones en su respectivo sector económico, ya sea primario, secundario o terciario.

Un factor importante para considerar es el tamaño de la empresa, que puede ser pequeña, mediana o grande, lo cual impacta significativamente en su estructura organizativa, recursos disponibles y capacidades operativas. Este análisis integral, que abarca desde el sector económico hasta la actividad de la organización, proporcionará una base sólida y valiosa para el desarrollo del programa.

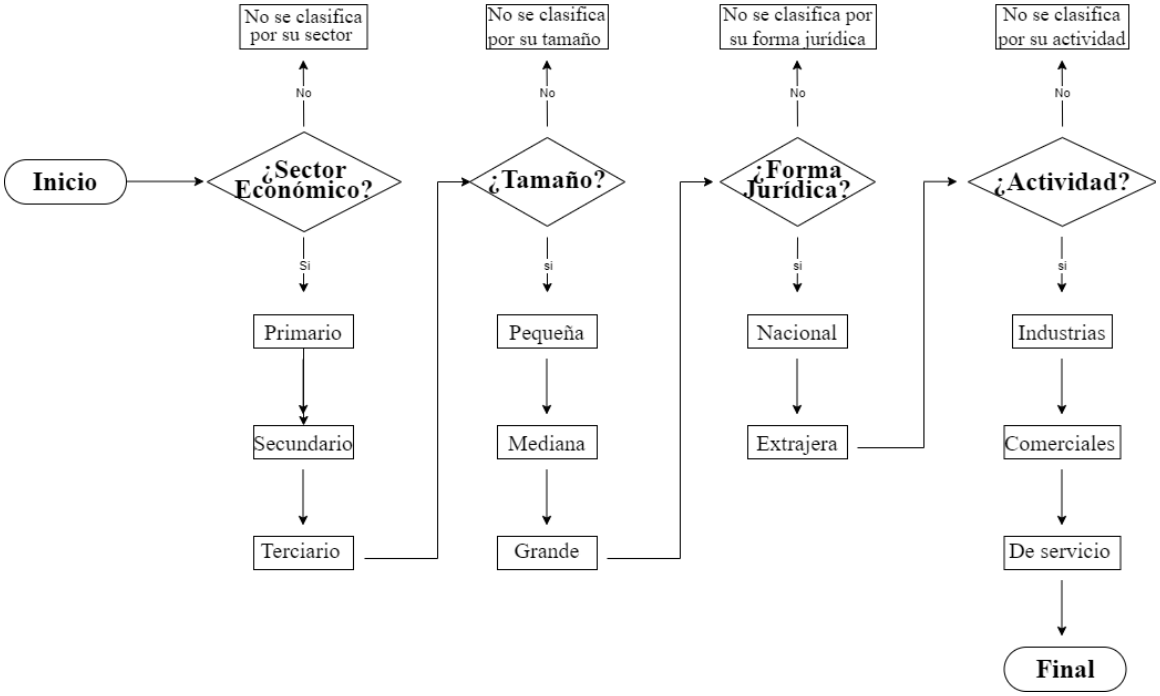


Figura 2. Flujograma tipos de empresas

El flujograma presentado en la Figura 2 muestra una categorización exhaustiva de las características y tipos de empresas existentes. Esta clasificación proporciona una comprensión más profunda de la diversidad empresarial, lo cual es crucial para realizar análisis sólidos y comprender el comportamiento y las variantes que puede exhibir una organización. Tener en cuenta estos parámetros permite abordar de

manera más efectiva las necesidades específicas de cada empresa según su naturaleza y circunstancias particulares.

La investigación se delimitó a las empresas con un sector económico primario. Estas compañías se caracterizan por contar con un número limitado de empleados y recursos, siendo organizaciones establecidas dentro del territorio nacional que realizan actividades comerciales relacionadas con la compra, venta y aprovisionamiento de productos. Debido a su tamaño y alcance, estas empresas se consideran de pequeña escala y operan principalmente a nivel local o regional.

La gestión de inventarios es un proceso esencial para estas empresas, ya que les permite tomar decisiones informadas en cuanto a la gestión de sus existencias. Es importante considerar que cada organización posee características únicas que influyen en la selección del modelo de inventario más adecuado. Estas características distintivas incluyen factores como el tipo de productos que manejan, los patrones de demanda de los clientes, los tiempos de entrega de los proveedores y los costos asociados al mantenimiento del inventario, entre otros aspectos relevantes.

En este sentido, es fundamental comprender que no existe un modelo de inventario universal que pueda ser plenamente aplicable a todas las empresas. Cada organización es única y enfrenta desafíos específicos en cuanto a la gestión de sus inventarios. Es necesario analizar y evaluar detenidamente las diversas alternativas disponibles, considerando los detalles y particularidades propias de cada empresa. Solo después de un análisis exhaustivo será posible seleccionar el modelo de inventario que mejor se ajuste a las necesidades y características específicas de la organización en cuestión. Este proceso de evaluación y selección es esencial para optimizar los recursos, minimizar los costos y maximizar la eficiencia en la gestión de las existencias, lo que a su vez contribuye al éxito general de la empresa.

Para la realización de la Tabla 7, se tomó en consideración los puntos más importantes de las fichas de investigación realizadas en el Anexo 4.

En la Tabla 7, se presentan los modelos más comunes que se conocieron en las entrevistas y los demás modelos existentes.

Tabla 7. Modelos de inventarios

Modelos de inventario		Descripción	Aplicación	Demanda
Modelos estáticos de cantidad económica de pedido (CEP, o *EOQ*)	**Modelo clásico de cantidad económica de pedido**	El modelo de inventario más básico asume una demanda constante y una reposición inmediata, lo que implica que los productos están disponibles de forma instantánea tras efectuarse el pedido.	Empresas con demanda estable de productos.	Determinista
	Cantidad económica de pedido con discontinuidades de precio	Este modelo es una variante del Modelo clásico de cantidad económica de pedido, pero con la diferencia de que permite obtener descuentos en el precio unitario de compra si el tamaño del pedido supera un límite específico.	Empresas que negocian descuentos por volumen con sus proveedores.	Determinista
	Cantidad económica de pedido de varios artículos con limitación de almacén	Este modelo se utiliza en casos con múltiples artículos (n > 1) cuyo nivel de *stock* varía según un patrón específico, sin permitir faltantes. La particularidad es que los artículos deben compartir un espacio de almacenamiento limitado, compitiendo por el mismo.	Empresas con espacios de almacenamiento limitados	Determinista
Modelos dinámicos de cantidad económica de pedido	**Modelo sin costo de preparación**	Este modelo considera un horizonte de planificación dividido en N períodos de igual duración y capacidad de producción limitada en cada uno, con varios niveles de producción (como tiempo	Empresas con costos de preparación de pedidos bajos	Determinista

Modelos de inventario		Descripción	Aplicación	Demanda
		normal y tiempo extra). Si en un periodo se produce más de lo demandado, el exceso se almacena para periodos posteriores, incurriendo en un costo de almacenamiento.		
Modelos de revisión continua	**Modelo con preparación**	En esta situación, no se admiten faltantes y se genera un costo de configuración cada vez que se inicia un lote de producción. Se propondrán dos enfoques de solución: un algoritmo exacto basado en programación dinámica y un algoritmo heurístico.	Empresas con costos de preparación de pedidos altos	Determinista
	Modelo "Probabilizado" de cantidad económica de pedido	Algunos profesionales han adaptado el modelo determinístico de cantidad económica de pedido para reflejar la naturaleza probabilista de la demanda, añadiendo una reserva constante al nivel de inventario durante todo el horizonte de planeación. Esta reserva se dimensiona para que la probabilidad de agotarse durante el tiempo de entrega no supere un valor especificado.	Empresas con demanda aleatoria, pero con un comportamiento predecible	Probabilístico
	Modelo probabilístico de cantidad	Permite la ocurrencia de faltantes durante la demanda y emplea una política de reorden en la que se realiza un	Empresas con demanda aleatoria y tiempos	Probabilístico

Modelos de inventario		Descripción	Aplicación	Demanda
	económica de pedido	pedido cuando el inventario alcanza un nivel R. Los valores óptimos de la cantidad a pedir (y) y el punto de reorden (R) se establecen minimizando el costo esperado por unidad de tiempo, que abarca los costos de preparación, almacenamiento y faltantes.	de entrega variables	
Modelos de un periodo	**Modelo sin preparación**	En este modelo, la demanda ocurre de manera instantánea al inicio del período, justo después de recibir el pedido. No se generan costos de preparación.	Empresas que realizan compras puntuales de productos.	Probabilístico
	Modelo con preparación (política s-S)	Este modelo varía del presentado en la sección "Modelo sin preparación" debido a la inclusión de un costo de preparación K. Usando la misma notación, se determina el costo total esperado por período.	Empresas que necesitan mantener un cierto nivel de inventario de seguridad para evitar faltantes de *stock*.	Probabilístico
	Modelos de varios periodos	Este modelo de varios períodos asume la ausencia de costos de preparación y permite que la demanda se acumule sin retrasos en la entrega. La demanda de cada período se modela con una función de distribución de probabilidades estacionaria.	Empresas con demanda y costos estacionales.	Probabilístico

Nota: Taha (2004).

Se analizaron todos los posibles modelos de gestión de inventarios que las empresas pueden adoptar. Este análisis busca contemplar diversos escenarios que una empresa podría enfrentar, tanto bajo modelos probabilísticos como determinísticos. El objetivo es obtener un diagnóstico preciso de las limitaciones y ventajas de cada modelo, para así considerar adecuadamente aquellos que sean relevantes en la creación del programa.

Después de estudiar y analizar todos los diferentes modelos de inventario, es importante también examinar todos los parámetros considerados en cada modelo. Aunque muchos modelos pueden ser determinísticos o probabilísticos, esto no implica que utilicen los mismos datos o parámetros. Por lo tanto, se realizó un análisis exhaustivo de todos los parámetros utilizados en cada modelo de inventario. Esto permitirá identificar qué parámetros son empleados por cada modelo, información que será fundamental al desarrollar el código del programa y al construir los árboles de decisiones. En la Tabla 8 se presentan los diferentes parámetros que utiliza cada modelo de inventario.

Tabla 8. Parámetros de modelos de inventario

Parámetros	**Modelos estáticos de cantidad económica de pedido (CEP, o *EOQ*)**			**Modelos dinámicos de cantidad económica de pedido**		**Modelos de revisión continua**		**Modelos de un periodo**		**Modelos de varios periodos**
	Modelo clásico de cantidad económica de pedido	Cantidad económica de pedido con discontinuidades de precio	Cantidad económica de pedido de varios artículos con limitación de almacén	Modelo sin costo de preparación	Modelo con preparación	Modelo "Probabilizado" de cantidad económica de pedido	Modelo probabilístico de cantidad económica de pedido	Modelo sin preparación	Modelo con preparación (política s-S)	
Cantidad pedida	✓	✓	✓	□	✓	□	✓	✓	✓	✓
Tasa de demanda	✓	✓	✓	□	✓	✓	✓	□	□	□
Costo de preparación de pedido	✓	✓	✓	□	✓	□	✓	✓	✓	□
Oferta acumulada	□	□	□	✓	□	□	□	□	□	□

Parámetros	Modelos estáticos de cantidad económica de pedido (CEP, o *EOQ*)			Modelos dinámicos de cantidad económica de pedido		Modelos de revisión continua		Modelos de un periodo		Modelos de varios periodos
	Modelo clásico de cantidad económica de pedido	Cantidad económica de pedido con discontinuidades de precio	Cantidad económica de pedido de varios artículos con limitación de almacén	Modelo sin costo de preparación	Modelo con preparación	Modelo "Probabilizado" de cantidad económica de pedido	Modelo probabilístico de cantidad económica de pedido	Modelo sin preparación	Modelo con preparación (política s-S)	
Demanda acumulada	☐	☐	☐	✓	☐	☐	☐	☐	☐	☐
Costo unitario de almacenamiento por unidad de tiempo	✓	✓	✓	☐	✓	☐	✓	☐	☐	☐
Tiempo de entrega	✓	✓	✓	☐	☐	✓	☐	☐	☐	☐

Parámetros	**Modelos estáticos de cantidad económica de pedido (CEP, o *EOQ*)**			**Modelos dinámicos de cantidad económica de pedido**		**Modelos de revisión continua**		**Modelos de un periodo**		**Modelos de varios periodos**
	Modelo clásico de cantidad económica de pedido	Cantidad económica de pedido con discontinuidades de precio	Cantidad económica de pedido de varios artículos con limitación de almacén	Modelo sin costo de preparación	Modelo con preparación	Modelo "Probabilizado" de cantidad económica de pedido	Modelo probabilístico de cantidad económica de pedido	Modelo sin preparación	Modelo con preparación (política s-S)	
Tamaño del pedido con descuento	☐	✓	☐	☐	☐	☐	☐	☐	☐	☐
Costo normal del pedido	☐	✓	☐	☐	☐	☐	☐	☐	☐	☐
Costo con descuento del pedido	☐	✓	☐	☐	☐	☐	☐	☐	☐	☐

Parámetros	Modelos estáticos de cantidad económica de pedido (CEP, o *EOQ*)			Modelos dinámicos de cantidad económica de pedido		Modelos de revisión continua		Modelos de un periodo		Modelos de varios periodos
	Modelo clásico de cantidad económica de pedido	Cantidad económica de pedido con discontinuidades de precio	Cantidad económica de pedido de varios artículos con limitación de almacén	Modelo sin costo de preparación	Modelo con preparación	Modelo "Probabilizado" de cantidad económica de pedido	Modelo probabilístico de cantidad económica de pedido	Modelo sin preparación	Modelo con preparación (política s-S)	
Área de almacenamiento necesaria por unidad de inventario	☐	☐	✓	☐	☐	☐	☐	☐	☐	☐
Área máxima disponible de almacenamiento para los n artículos	☐	☐	✓	☐	☐	☐	☐	☐	☐	☐
Demanda para periodo i	☐	☐	☐	☐	✓	☐	☐	☐	☐	☐

Parámetros	**Modelos estáticos de cantidad económica de pedido (CEP, o *EOQ*)**			**Modelos dinámicos de cantidad económica de pedido**		**Modelos de revisión continua**		**Modelos de un periodo**		**Modelos de varios periodos**
	Modelo clásico de cantidad económica de pedido	Cantidad económica de pedido con discontinuidades de precio	Cantidad económica de pedido de varios artículos con limitación de almacén	Modelo sin costo de preparación	Modelo con preparación	Modelo "Probabilizado" de cantidad económica de pedido	Modelo probabilístico de cantidad económica de pedido	Modelo sin preparación	Modelo con preparación (política s-S)	
Inventario al inicio del periodo i	☐	☐	☐	☐	✓	☐	☐	☐	☐	☐
Demanda esperada por unidad de tiempo	☐	☐	☐	☐	☐	☐	✓	☐	☐	☐
Costo de almacenamiento por unidad de inventario y	☐	☐	✓	☐	☐	☐	✓	☐	☐	☐

Parámetros	Modelos estáticos de cantidad económica de pedido (CEP, o *EOQ*)			Modelos dinámicos de cantidad económica de pedido		Modelos de revisión continua		Modelos de un periodo		Modelos de varios periodos
	Modelo clásico de cantidad económica de pedido	Cantidad económica de pedido con discontinuidades de precio	Cantidad económica de pedido de varios artículos con limitación de almacén	Modelo sin costo de preparación	Modelo con preparación	Modelo "Probabilizado" de cantidad económica de pedido	Modelo probabilístico de cantidad económica de pedido	Modelo sin preparación	Modelo con preparación (política s-S)	
por unidad de tiempo										
Costo de faltante por unidad de inventario	☐	☐	☐	☐	☐	☐	✓	☐	☐	☐
Función de distribución de probabilidades de la demanda x durante el	☐	☐	☐	☐	☐	☐	✓	☐	☐	☐

Parámetros	Modelos estáticos de cantidad económica de pedido (CEP, o *EOQ*)			Modelos dinámicos de cantidad económica de pedido		Modelos de revisión continua		Modelos de un periodo		Modelos de varios periodos
	Modelo clásico de cantidad económica de pedido	Cantidad económica de pedido con discontinuidades de precio	Cantidad económica de pedido de varios artículos con limitación de almacén	Modelo sin costo de preparación	Modelo con preparación	Modelo "Probabilizado" de cantidad económica de pedido	Modelo probabilístico de cantidad económica de pedido	Modelo sin preparación	Modelo con preparación (política s-S)	
tiempo de entrega										
Costo de compra por unidad	☐	☐	☐	☐	☐	☐	☐	✓	✓	✓
Costo de almacenamiento por unidad	☐	☐	☐	☐	☐	☐	☐	✓	✓	✓

Parámetros	Modelos estáticos de cantidad económica de pedido (CEP, o *EOQ*)			Modelos dinámicos de cantidad económica de pedido		Modelos de revisión continua		Modelos de un periodo		Modelos de varios periodos
	Modelo clásico de cantidad económica de pedido	Cantidad económica de pedido con discontinuidades de precio	Cantidad económica de pedido de varios artículos con limitación de almacén	Modelo sin costo de preparación	Modelo con preparación	Modelo "Probabilizado" de cantidad económica de pedido	Modelo probabilístico de cantidad económica de pedido	Modelo sin preparación	Modelo con preparación (política s-S)	
Penalización por unidad faltante durante el periodo	☐	☐	☐	☐	☐	☐	☐	✓	✓	✓
Cantidad a la mano, antes de hacer un pedido	☐	☐	☐	☐	☐	☐	☐	✓	✓	✓
Ingreso por unida	☐	☐	☐	☐	☐	☐	☐	☐	☐	✓

Parámetros	**Modelos estáticos de cantidad económica de pedido (CEP, o *EOQ*)**			**Modelos dinámicos de cantidad económica de pedido**		**Modelos de revisión continua**		**Modelos de un periodo**		**Modelos de varios periodos**
	Modelo clásico de cantidad económica de pedido	Cantidad económica de pedido con discontinuidades de precio	Cantidad económica de pedido de varios artículos con limitación de almacén	Modelo sin costo de preparación	Modelo con preparación	Modelo "Probabilizado" de cantidad económica de pedido	Modelo probabilístico de cantidad económica de pedido	Modelo sin preparación	Modelo con preparación (política s-S)	
Precio de venta por unidad	☐	☐	☐	☐	☐	☐	☐	☐	☐	✓
Factor de descuento	☐	☐	☐	☐	☐	☐	☐	☐	☐	✓
Tiempo de entrega entre la colocación y la recepción	☐	☐	☐	☐	☐	✓	☐	☐	☐	☐

Parámetros	**Modelos estáticos de cantidad económica de pedido (CEP, o *EOQ*)**			**Modelos dinámicos de cantidad económica de pedido**		**Modelos de revisión continua**		**Modelos de un periodo**		**Modelos de varios periodos**
	Modelo clásico de cantidad económica de pedido	Cantidad económica de pedido con discontinuidades de precio	Cantidad económica de pedido de varios artículos con limitación de almacén	Modelo sin costo de preparación	Modelo con preparación	Modelo "Probabilizado" de cantidad económica de pedido	Modelo probabilístico de cantidad económica de pedido	Modelo sin preparación	Modelo con preparación (política s-S)	
Variable aleatoria que representa la demanda durante el tiempo de entrega	☐	☐	☐	☐	☐	✓	☐	☐	☐	☐
Demanda promedio durante el tiempo de entrega	☐	☐	☐	☐	☐	✓	☐	☐	☐	☐

Parámetros	Modelos estáticos de cantidad económica de pedido (CEP, o *EOQ*)			Modelos dinámicos de cantidad económica de pedido		Modelos de revisión continua		Modelos de un periodo		Modelos de varios periodos
	Modelo clásico de cantidad económica de pedido	Cantidad económica de pedido con discontinuidades de precio	Cantidad económica de pedido de varios artículos con limitación de almacén	Modelo sin costo de preparación	Modelo con preparación	Modelo "Probabilizado" de cantidad económica de pedido	Modelo probabilístico de cantidad económica de pedido	Modelo sin preparación	Modelo con preparación (política s-S)	
Desviación estándar de la demanda durante el tiempo de entrega	☐	☐	☐	☐	☐	✓	☐	☐	☐	☐
Tamaño de la existencia de reserva	☐	☐	☐	☐	☐	✓	☐	☐	☐	☐
Probabilidad máxima admisible de	☐	☐	☐	☐	☐	✓	☐	☐	☐	☐

Parámetros	Modelos estáticos de cantidad económica de pedido (CEP, o *EOQ*)			Modelos dinámicos de cantidad económica de pedido		Modelos de revisión continua		Modelos de un periodo		Modelos de varios periodos
	Modelo clásico de cantidad económica de pedido	Cantidad económica de pedido con discontinuidades de precio	Cantidad económica de pedido de varios artículos con limitación de almacén	Modelo sin costo de preparación	Modelo con preparación	Modelo "Probabilizado" de cantidad económica de pedido	Modelo probabilístico de cantidad económica de pedido	Modelo sin preparación	Modelo con preparación (política s-S)	
que se agote la existencia durante el tiempo de entrega										
Variable aleatoria que representa la demanda durante el periodo	☐	☐	☐	☐	☐	☐	☐	✓	✓	✓

Parámetros	Modelos estáticos de cantidad económica de pedido (CEP, o *EOQ*)			Modelos dinámicos de cantidad económica de pedido		Modelos de revisión continua		Modelos de un periodo		Modelos de varios periodos
	Modelo clásico de cantidad económica de pedido	Cantidad económica de pedido con discontinuidades de precio	Cantidad económica de pedido de varios artículos con limitación de almacén	Modelo sin costo de preparación	Modelo con preparación	Modelo "Probabilizado" de cantidad económica de pedido	Modelo probabilístico de cantidad económica de pedido	Modelo sin preparación	Modelo con preparación (política s-S)	
Costo de penalización por unidad y unidad de tiempo	☐	☐	☐	☐	☐	☐	☐	☐	☐	✓
Distribución de la función de probabilidad de la demanda durante el periodo	☐	☐	☐	☐	☐	☐	☐	✓	☐	✓

Fuente: (Taha, 2004).

4.1.3. Determinar el lenguaje más adecuado para la programación del sistema de gestión de inventarios.

En el desarrollo de un sistema de gestión de inventarios eficiente, la elección del lenguaje de programación adecuado es un factor demasiado importante que puede influir significativamente en el desempeño, la escalabilidad y la mantenibilidad del sistema. Este objetivo específico se centra en identificar el lenguaje de programación más apropiado para la implementación del programa de gestión de inventarios basado en un árbol de decisiones, con el fin de asegurar que el sistema no solo cumpla con los requisitos funcionales, sino que también optimice los recursos disponibles en las empresas.

La selección del lenguaje de programación debe considerar varios criterios esenciales, tales como la facilidad de uso, la compatibilidad con otras tecnologías, el soporte a largo plazo y la comunidad de desarrolladores. Además, es importante que sea lo más fácil posible la inscripción y utilización de árboles de decisión, que será la parte más importante del código, además debe ser fácilmente modificable según las necesidades y procesos que se identificaron anteriormente. En el presente capítulo, se explorarán y analizarán diferentes lenguajes de programación populares, comparando sus características y beneficios en relación con los requisitos específicos del sistema de gestión de inventarios. A través de este análisis, se determinará cuál es el lenguaje más adecuado para garantizar una implementación eficiente y efectiva del sistema, contribuyendo así a la optimización. En la Tabla 9 se analizan los lenguajes de programación más utilizados.

Tabla 9. Lenguajes de programación más utilizados

Lenguaje	**Plataforma**	**Curva de aprendizaje**	**Bibliotecas y *Frameworks***	**Rendimiento**	**Integración con base de datos**
Java	Multiplataforma	Moderada	*Spring, Hibernate.*	Alto rendimiento	Buena integración con varias Base de Datos
Python	Multiplataforma	Fácil	*Django, flask, Pandas, Numpy, Scikit-learn.*	Rendimiento moderado	Buena integración con varias Base de Datos
C#	*Windows, .NET*	Moderada	*ASP.NET, Entity Framework.*	Alto rendimiento	Excelente integración con SQL Server

Lenguaje	Plataforma	Curva de aprendizaje	Bibliotecas y *Frameworks*	Rendimiento	Integración con base de datos
PHP	Multiplataforma (principalmente *web*)	Fácil	*Laravel, Symfony*	Rendimiento moderado	Buena integración con MySQL, PostgreSQL
JavaScript (Node.js)	Multiplataforma (principalmente *web*)	Moderada	*Experess, React, Angular*	Rendimiento moderado	Buena integración con MongoDB, PostgreSQL

Fuente: Stackscale (2023).

Se elaboró la Tabla 9, con los tipos de lenguajes más populares en el mundo con las distintas características que se buscan, para así determinar cuál es el más adecuado para el programa. Después de un análisis exhaustivo de estos lenguajes de programación, se determinó que *Python* es el lenguaje más adecuado para la programación del sistema de gestión de inventarios basado en un árbol de decisiones.

La elección de *Python* se fundamenta en varios aspectos clave, Redondo (2019) indica que *Python* es conocido por su facilidad de uso y legibilidad, características que facilitan tanto el desarrollo inicial como el mantenimiento del código. Su sintaxis clara y sencilla permite a los desarrolladores centrarse en la lógica del negocio sin preocuparse por la complejidad del lenguaje, lo que reduce la probabilidad de errores y acelera el proceso de desarrollo. Esta simplicidad es particularmente ventajosa para un sistema de gestión de inventarios, donde la precisión y la eficiencia son fundamentales.

Además, McKinney (2010) ya avisa de que *Python* ofrece una amplia gama de bibliotecas especializadas para el análisis de datos y la implementación de algoritmos de aprendizaje automático, como *NumPy*, *pandas* y *scikit-learn* (pág. 60). Estas herramientas son esenciales para desarrollar un algoritmo de árbol de decisiones eficiente y para gestionar grandes volúmenes de datos de inventario. La robustez y versatilidad de estas bibliotecas permiten un análisis de datos más profundo y preciso, contribuyendo significativamente a la optimización de los recursos empresariales.

La capacidad de *Python* para integrarse con otros sistemas y tecnologías es otro factor crucial en su selección Bayer (2022). *Python* puede interactuar fácilmente con

bases de datos, servicios *web* y otras aplicaciones empresariales mediante bibliotecas como *SQLAlchemy*, *Django* y *Flask*. Esta compatibilidad es vital para un sistema de gestión de inventarios que necesita comunicarse con otros módulos del sistema empresarial, asegurando una operación fluida y coherente dentro de la infraestructura tecnológica de la empresa.

Python también se beneficia de una de las comunidades de desarrolladores más grandes y activas del mundo Redondo (2019). Esta extensa comunidad proporciona acceso a una vasta cantidad de recursos, tutoriales, documentación y soporte técnico, lo cual es invaluable durante el desarrollo y la implementación del sistema. La comunidad activa también garantiza que el lenguaje evolucione y se mantenga actualizado con las últimas tendencias y tecnologías, ofreciendo una plataforma robusta y de vanguardia para el desarrollo de *software*.

Python emerge como el lenguaje de programación ideal para desarrollar el sistema de gestión de inventarios basado en un árbol de decisiones. Sus ventajas en términos de facilidad de uso, herramientas de análisis de datos, capacidades de integración, soporte comunitario, rendimiento, seguridad y adopción empresarial lo convierten en la opción más adecuada. Estas características aseguran que el sistema sea no solo eficiente y efectivo, sino también escalable y seguro. La elección de *Python* permitió desarrollar el proyecto de manera más sencilla y eficaz, cumpliendo con las especificaciones deseadas. Específicamente, se necesitaba un sistema que se adaptara a los procesos identificados en las empresas de Tulcán y que fuera de fácil uso para los usuarios inexpertos. En este sentido, *Python* se destacó como el lenguaje más adecuado para cumplir los objetivos, proporcionando una solución robusta y accesible que responde a las necesidades prácticas del entorno empresarial local.

4.1.4. Diseñar y desarrollar un programa para la gestión de inventarios

Aunque en los objetivos anteriores de este trabajo de titulación no se evidenció una gran cantidad de investigación y esfuerzo, en este objetivo se demuestra toda la investigación y trabajo realizados para su finalización.

Para el diseño y desarrollo del programa de gestión de inventarios, es necesario recopilar toda la información obtenida de los objetivos anteriores. En el primer objetivo se determinó las actividades específicas que realizan las distintas empresas de Tulcán, las cuales serán el enfoque principal del programa. Una vez definidas estas

actividades, el segundo objetivo consistió en evaluar los diferentes tipos de modelos de inventarios existentes. Aunque las empresas de Tulcán utilizaban el modelo más básico de inventarios, el *EOQ*, se estableció como meta que el programa pudiera manejar cualquier tipo de modelo de inventario, ya sea determinístico o probabilístico. Finalmente, se determinó que el lenguaje de programación apropiado para el desarrollo del programa era *Python*, debido a sus múltiples ventajas y su sencillez al programar.

4.1.4.1. Flujogramas del programa

Una vez alcanzados los objetivos previamente establecidos, se identificó de manera precisa las áreas clave en la gestión de inventarios. Sin embargo, como se destacó en la justificación de este trabajo, el propósito central no se limita a realizar cálculos sencillos de entradas y salidas, ya que tales operaciones son fácilmente manejables por cualquier *software* convencional. En lugar de eso, este estudio se enfoca en un aspecto más avanzado y desafiante: la implementación de modelos determinísticos y probabilísticos para la optimización del manejo de inventarios.

El verdadero valor de este enfoque radica en la capacidad para calcular de manera eficiente la cantidad óptima de pedido y el punto de reorden, dos variables críticas que permiten abordar de manera efectiva la mayoría de los problemas encontrados en la administración de inventarios. Al determinar estos dos elementos clave, no solo se optimizan los costos asociados al inventario, sino que también se asegura la disponibilidad continua de productos, evitando tanto el exceso de *stock* como los costosos desabastecimientos. Además, estos modelos permiten una planificación más precisa y adaptable a las fluctuaciones de demanda, lo cual es particularmente importante en entornos inciertos o altamente volátiles.

En primer lugar, fue importante determinar la lógica del programa para su desarrollo, es decir, establecer paso a paso qué va a realizar el programa. Para ilustrar mejor esta lógica, se crearon varios flujogramas que explican su funcionamiento. Se elaboró un flujograma general del programa, el cual muestra su funcionamiento global, pero no detalla las acciones específicas para cada modelo de inventarios. En los flujogramas siguientes se describe detalladamente la lógica de cada modelo de inventarios.

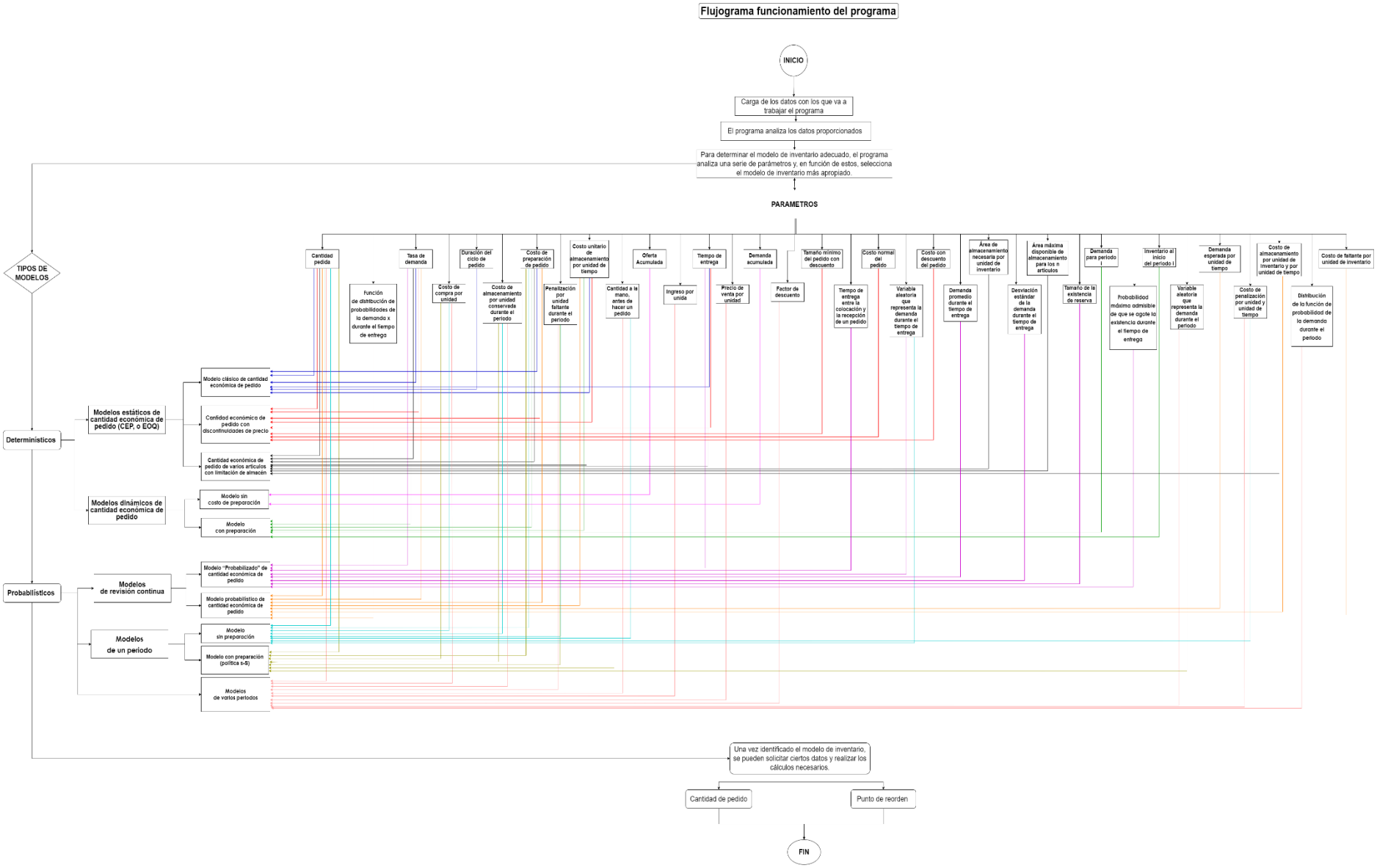


Figura 3. Flujograma funcionamiento del programa

En la Figura 3 se puede apreciar la lógica que seguirá el programa, representada en forma de flujograma. En este primer flujograma se determinó cómo funcionará el programa paso a paso, de la siguiente manera:

Paso 1. _ Los usuarios acceden al programa, donde podrán elegir entre los distintos modelos de inventario disponibles. El usuario seleccionará el modelo con el que desea trabajar.

Paso 2. _ Una vez seleccionado el modelo, se deberán ingresar los valores o parámetros necesarios para que el programa pueda realizar los cálculos correspondientes.

Paso 3. _ El programa analiza los datos proporcionados. El sistema comienza revisando la integridad y la coherencia de los datos cargados. Verifica que no haya valores faltantes o errores evidentes que puedan afectar el análisis posterior.

Paso 4. _ Cálculo de la cantidad a pedir y el punto de reorden. Una vez determinado el modelo específico de inventario, el programa calcula automáticamente la cantidad óptima a pedir y el punto de reorden correspondiente. Para cada modelo de inventario, sigue una lógica diferente basada en las fórmulas y criterios específicos de ese modelo. El resultado es una recomendación precisa y adaptada a las necesidades del usuario.

Este proceso garantiza que el programa pueda manejar eficazmente diversas situaciones de inventario, proporcionando soluciones óptimas basadas en los datos y modelos más adecuados. La lógica detallada en la ilustración 3 es fundamental para el funcionamiento del programa con todos los datos que se le proporcionen.

El programa sigue una serie de pasos claramente definidos para analizar los datos y determinar el modelo de inventario específico a aplicar. Este enfoque es aplicable a todos los modelos de inventarios que se han considerado. Según se observa en la Figura 3, el programa analiza minuciosamente los datos ingresados, identificando los parámetros relevantes. Basándose en estos parámetros, el programa desvía su operación hacia un modelo específico de inventario.

En total, el programa analiza 35 parámetros diferentes para identificar el modelo de inventario exacto que corresponde a los datos proporcionados. Esta capacidad de análisis detallado asegura que el programa pueda ofrecer recomendaciones precisas y adaptadas a las necesidades específicas del inventario.

Es importante destacar que esta fue solo la primera fase del desarrollo del programa. Aunque en este punto el programa operaba según la lógica del flujograma, en fases posteriores se entrenó al programa para que actúe de manera más sofisticada. Este avance, que se explica en las páginas posteriores de este trabajo, esto permitió al programa adaptarse y mejorar su desempeño en la gestión de inventarios.

Esta evolución en el diseño del programa refleja un enfoque exhaustivo y progresivo para garantizar que el sistema no solo cumpla con los requisitos actuales, sino que también sea capaz de adaptarse a futuros desafíos y necesidades en la gestión de inventarios. Con esta metodología, se ha asegurado que el programa no solo maneje eficazmente los datos actuales, sino que también esté preparado para incorporar nuevas funcionalidades y mejoras continuas, maximizando su utilidad y eficiencia a largo plazo.

4.1.4.2. Modelos determinísticos

En esta sección se evidencia la elaboración y explicación detallada de los flujogramas de únicamente los modelos determinísticos, resulta fundamental señalar que estos han sido desarrollados tomando como referencia el libro de Taha (2004). Taha. Este texto ha sido la principal fuente de información y consulta para todo el proyecto, proporcionando no solo las bases teóricas, sino también los cálculos necesarios para el correcto desarrollo de cada uno de los procesos.

La obra de Taha ha sido instrumental en la estructuración del proyecto, ya que ofrece un enfoque riguroso y metódico de los principios y herramientas de la investigación de operaciones, lo que ha permitido garantizar la precisión y validez de los resultados obtenidos. En este sentido, cada flujograma ha sido construido de manera meticulosa, siguiendo las metodologías y ejemplos expuestos en el mencionado libro. La obra de Taha proporciona un marco teórico sólido que abarca desde la formulación de problemas hasta la implementación de soluciones óptimas, lo cual fue determinante para la realización de los cálculos y la interpretación de los datos. Por tanto, se puede afirmar que todo el desarrollo de este proyecto se fundamenta en los conocimientos y técnicas extraídas directamente de esta fuente, garantizando un enfoque coherente y alineado con los estándares establecidos en la disciplina de investigación de operaciones.

4.1.4.2.1. Flujogramas modelo clásico de cantidad económica de pedido

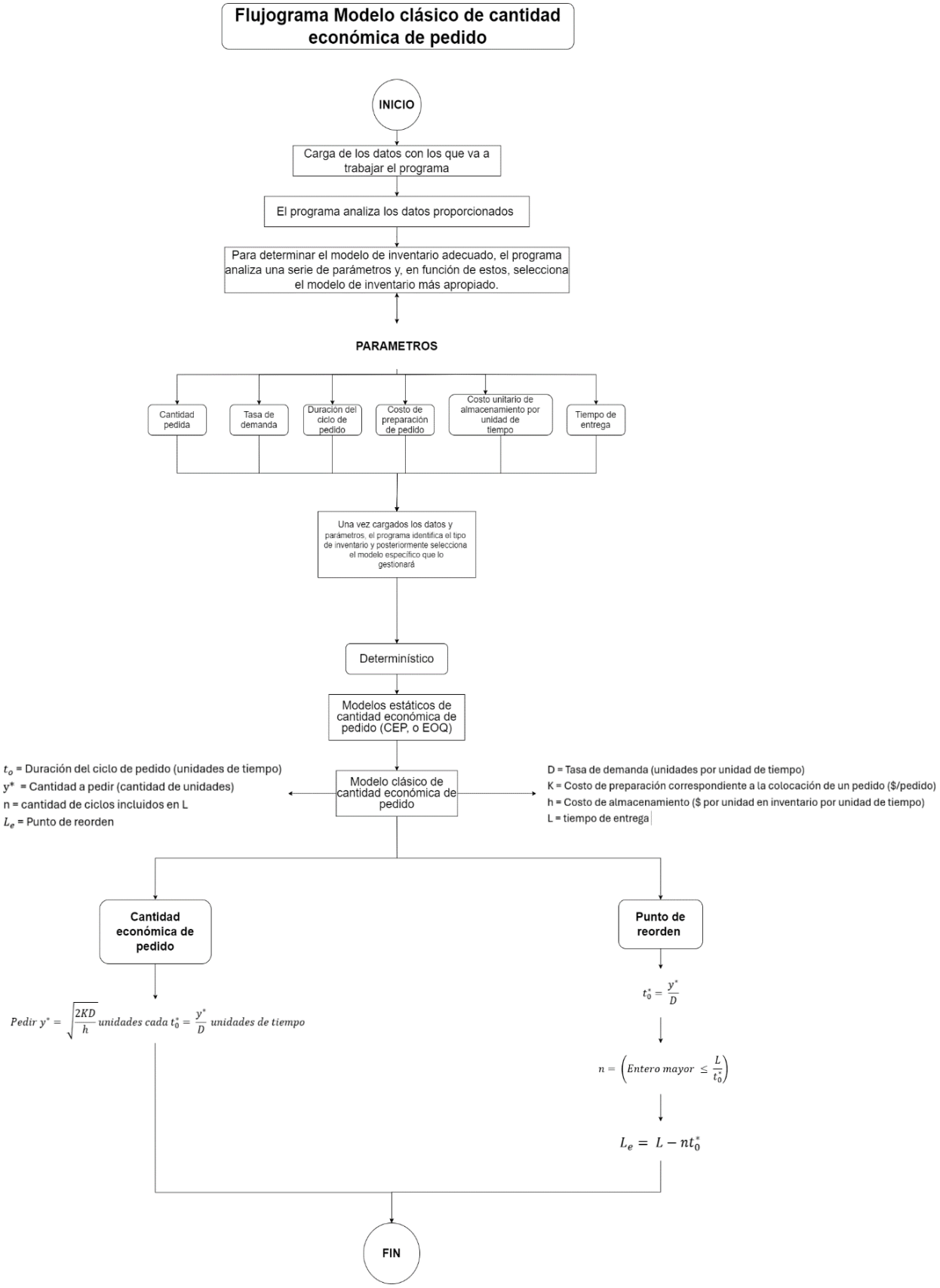


Figura 4. Flujograma modelo clásico de cantidad económica de pedido

En la Figura 4, se detalla cómo se realizarán los cálculos de la cantidad a pedir y del punto de reorden para este modelo específico. El flujograma muestra de manera clara y concisa cada paso del proceso, asegurando que los cálculos se realicen de manera precisa y eficiente.

Para que este modelo funcione correctamente, es esencial cargar los datos con los siguientes parámetros específicos:

D = Tasa de demanda (u/u de tiempo)

K = Costo de preparación correspondiente a la colocación de un pedido ($/pedido)

h = Costo de almacenamiento ($ por unidad en inventario por unidad de tiempo)

L = Tiempo de entrega

El programa identifica estos parámetros e inmediatamente se desvía hacia el modelo clásico de cantidad económica de pedido (*EOQ*). Una vez determinado que este es el modelo de inventario adecuado, el programa sigue una serie de pasos específicos para llevar a cabo los cálculos necesarios. Estos pasos aseguran que el proceso se realice de manera precisa y eficiente. A continuación, se describen los pasos que sigue el programa para este modelo en particular:

Paso 1. _ para el cálculo de la cantidad económica de pedido, el programa usa los parámetros K, D y h. Cuando el programa identifica estos parámetros hace uso de la formula:

$$\boldsymbol{y^*} = \sqrt{\frac{\boldsymbol{2KD}}{\boldsymbol{h}}} \tag{6}$$

Esta fórmula permite al programa determinar la cantidad ideal de producto que se debe solicitar para minimizar los costos totales de inventario.

Paso 2. _ Una vez determinada la cantidad a pedir, el programa hace el cálculo de en qué tiempo se deben pedir estas cantidades de pedido, para esto hace uso de los parámetros D y la reciente cantidad de pedido calculada y^*, en la formula:

$$\boldsymbol{t^*_o} = \frac{\boldsymbol{y^*}}{\boldsymbol{D}} \tag{7}$$

Esta fórmula permite al programa determinar el intervalo de tiempo ideal entre pedidos. Al aplicar esta fórmula, el programa optimiza el momento para realizar nuevos pedidos, asegurando una gestión eficiente del inventario y minimizando tanto los costos de almacenamiento como la posibilidad de agotar existencias.

Paso 3. _ Para calcular el punto de reorden, en primer lugar, el programa calcula la cantidad de ciclos en L, con la siguiente fórmula:

$$\boldsymbol{n = (Entero\ mayor\ \leq \frac{L}{t^*_o})} \quad (8)$$

Una vez calculado n, el programa hace uso de otra fórmula para saber el punto de reorden exacto, en este caso la fórmula es:

$$\boldsymbol{L_e = L - nt^*_0} \quad (9)$$

Este el punto de reorden calculado, que indica el momento exacto en el cual debe realizarse un nuevo pedido para evitar la escasez de inventario. Al aplicar estas fórmulas, el programa garantiza una gestión precisa del inventario, asegurando que los pedidos se realicen de manera oportuna para mantener niveles adecuados de *stock* y evitar interrupciones en la disponibilidad de productos.

4.1.4.2.2. Flujograma modelo de cantidad económica de pedido con discontinuidades de precio

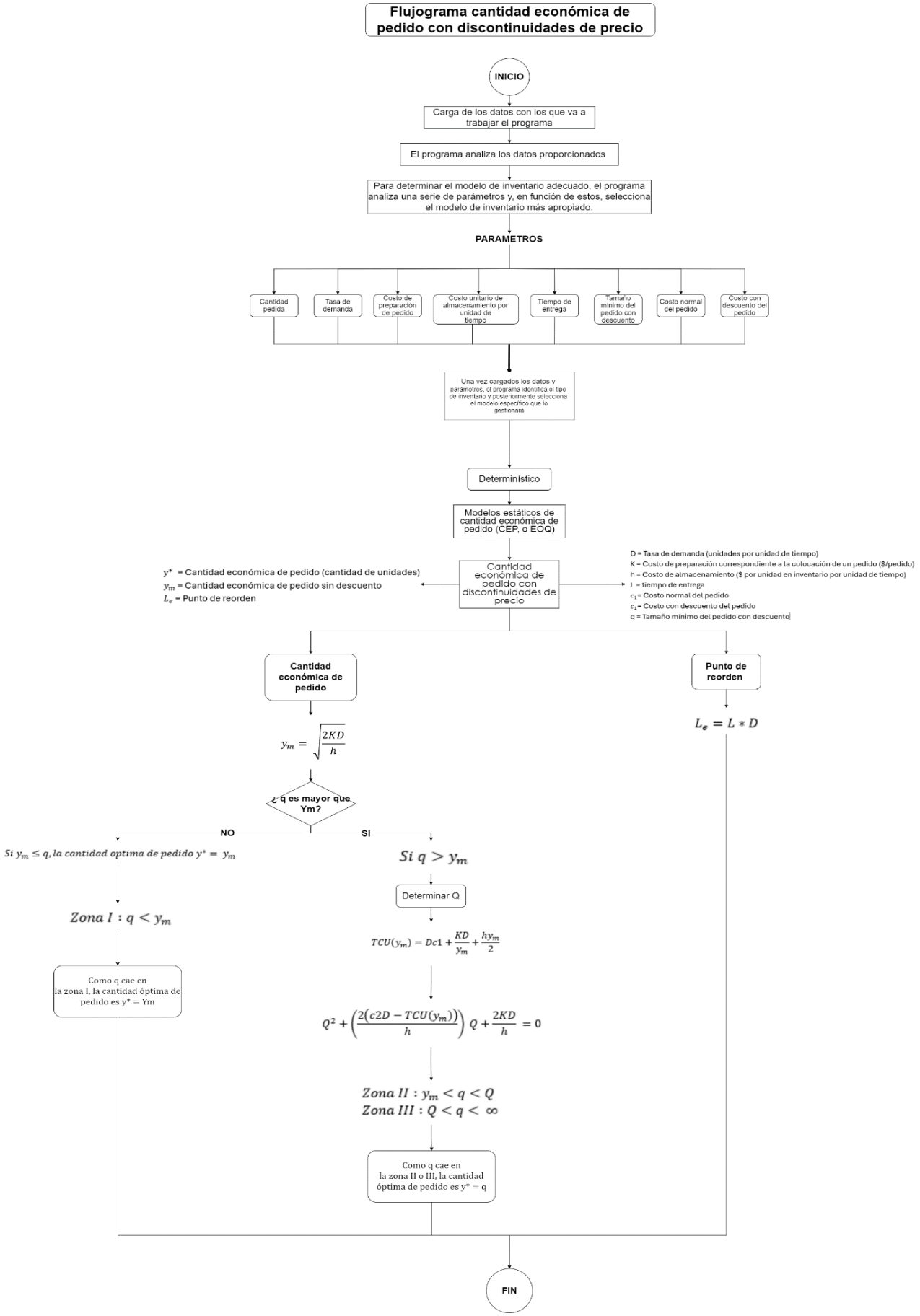


Figura 5. Flujograma modelo de cantidad económica de pedido con discontinuidades de precio

El siguiente modelo que el programa puede analizar es el modelo de cantidad económica de pedido con discontinuidades de precio, su lógica se puede visualizar en la Figura 5.

Para que este modelo funcione correctamente, es esencial cargar los datos con los siguientes parámetros específicos:

D = Tasa de demanda (u/u de tiempo)

K = Costo de preparación correspondiente a la colocación de un pedido ($/pedido)

h = Costo de almacenamiento ($ por unidad en inventario por unidad de tiempo)

L = Tiempo de entrega

$\boldsymbol{c_1}$ **=** Costo normal del pedido

$\boldsymbol{c_2}$ **=** Costo con descuento del pedido

q = tamaño mínimo del pedido con descuento

El programa identifica estos parámetros e inmediatamente se desvía hacia el modelo de cantidad económica de pedido con discontinuidades de precios. Una vez determinado que este es el modelo de inventario adecuado, el programa sigue una serie de pasos específicos para llevar a cabo los cálculos necesarios. Estos pasos aseguran que el proceso se realice de manera precisa y eficiente. A continuación, se describen los pasos que sigue el programa para este modelo en particular:

Paso 1. _ para el cálculo de la cantidad económica de pedido, el programa usa los parámetros K, D y h. Cuando el programa identifica estos parámetros hace uso de la formula:

$$\boldsymbol{y_m} = \sqrt{\frac{\boldsymbol{2KD}}{\boldsymbol{h}}} \qquad (10)$$

Esta fórmula permite al programa determinar la cantidad ideal de producto que se debe solicitar para minimizar los costos totales de inventario.

Aquí es importante destacar que este cálculo de y_m es la cantidad económica de pedido, pero sin tomar en cuenta el descuento.

A partir de este punto, el programa puede tomar una de dos decisiones según una variante específica, basada en la pregunta: ¿q > y_m?.

Si q > y_m, entonces:

Paso 1. _ Para proceder con esta variante, el programa primero calcula QQ utilizando la siguiente fórmula para la función de costos total unitario (*TCU*):

$$\boldsymbol{TCU(y_m) = Dc1 + \frac{KD}{y_m} + \frac{hy_m}{2}} \quad (11)$$

Paso 2. _ Una vez calcula el *TCU*, ahora si el programa procede hacer uso de la siguiente fórmula para calcular Q.

$$\boldsymbol{Q^2 + \left(\frac{2\left(c2D - TCU(y_m)\right)}{h}\right) Q + \frac{2KD}{h} = 0} \quad (12)$$

Con esta fórmula, el programa calcula QQ. Posteriormente, se evalúan dos condicionantes:

$$Zona\ II : y_m < q < Q$$
$$Zona\ III : Q < q < \infty$$

Dependiendo de en qué zona se encuentre q respecto a Q:

- Si q está en la Zona II, el programa determina que la cantidad económica de pedido es q.
- Si q cae en la Zona III, el programa determina que la cantidad económica de pedido es y_m.

Si q ≤ y_m, entonces:

Paso 1. _ Dado que q es menor que y_m, el programa automáticamente determina que está en la Zona I, por lo tanto, la cantidad económica de pedido es y_m. En este caso, el programa no realiza ningún otro cálculo.

Una vez calculada la cantidad económica de pedido, el programa procede a calcular el punto de reorden de la siguiente manera:

Paso 1. _ Para el cálculo del punto de reorden el programa hace uso de la siguiente formula:

$$\boldsymbol{L_e = L * D} \quad (13)$$

Esta fórmula permite al programa calcular el nivel de inventario en el cual se debe realizar un nuevo pedido para evitar la escasez.

4.1.4.2.3. Flujograma modelo de cantidad económica de pedido de varios artículos con limitación de almacén

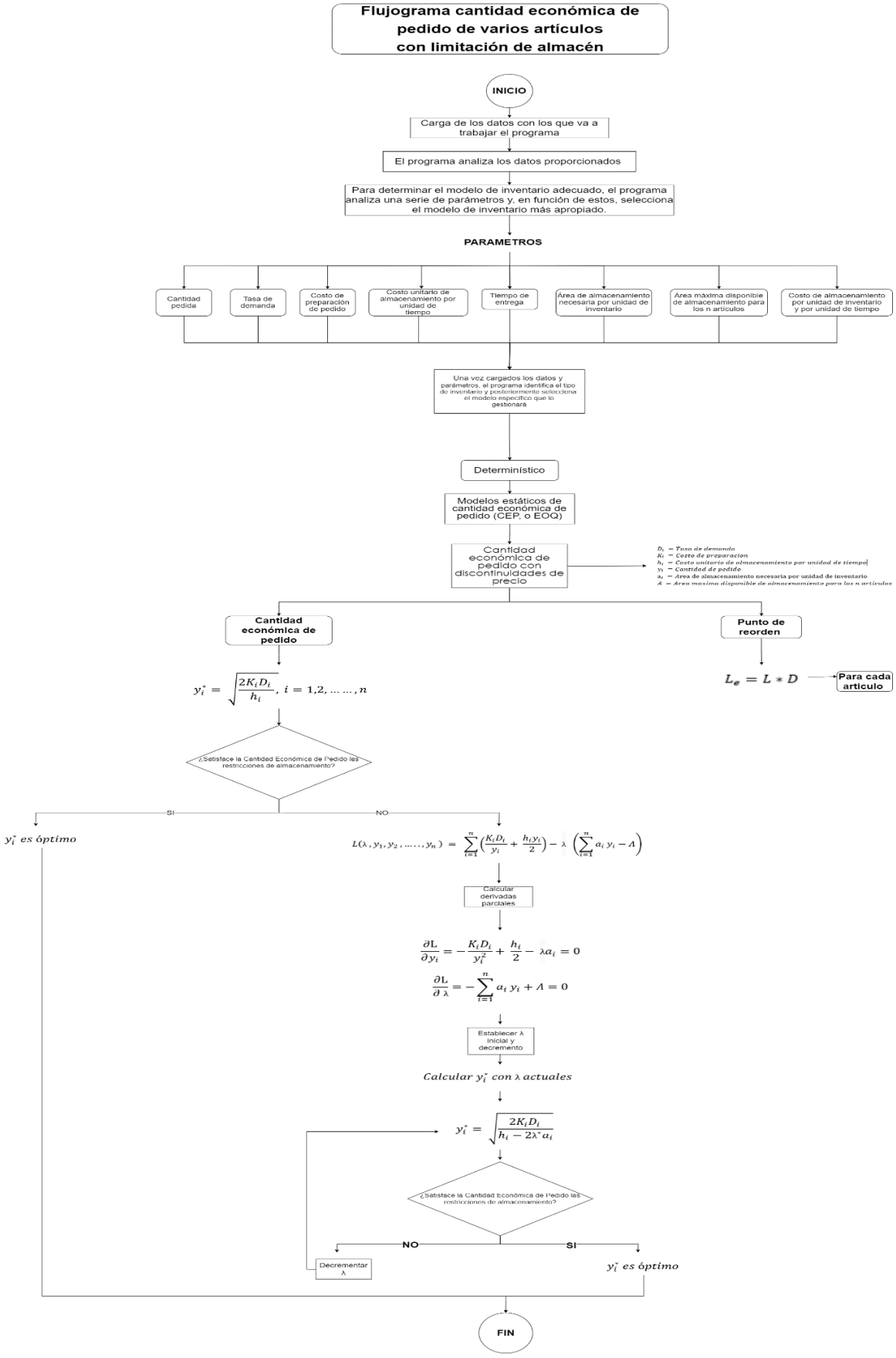


Figura 6. Flujograma modelo de cantidad económica de pedido de varios artículos con limitación de almacén

En la Figura 6, se ilustra la lógica que sigue el modelo de cantidad económica de pedido para varios artículos con limitación de almacén. La figura detalla todos los parámetros y la secuencia de procesos que este modelo sigue dentro del programa.

Para que este modelo funcione correctamente, es esencial cargar los datos con los siguientes parámetros específicos:

$\boldsymbol{D_i}$ =Tasa de demanda

$\boldsymbol{K_i}$ =Costo de preparación

$\boldsymbol{h_i}$ =Costo unitario de almacenamiento por unidad de tiempo

$\boldsymbol{y_i}$ =Cantidad de pedido

$\boldsymbol{a_i}$ =Área de almacenamiento necesaria por unidad de inventario

A =Área máxima disponible de almacenamiento para los n artículos

El programa identifica estos parámetros e inmediatamente se desvía hacia el modelo de cantidad económica de pedido de varios artículos con limitación de almacén. Una vez determinado que este es el modelo de inventario adecuado, el programa sigue una serie de pasos específicos para llevar a cabo los cálculos necesarios. Estos pasos aseguran que el proceso se realice de manera precisa y eficiente. A continuación, se describen los pasos que sigue el programa para este modelo en particular:

Paso 1. _ Para el primer paso en este modelo el programa calcula los valores óptimos no restringidos de las cantidades de pedido para los diferentes artículos, con la siguiente formula:

$$\boldsymbol{y^*_i} = \sqrt{\frac{\boldsymbol{2K_iD_i}}{\boldsymbol{h_i}}}\,, \boldsymbol{i = 1,, \ldots \ldots, n} \qquad (14)$$

A partir de este punto, el programa puede tomar una de dos decisiones según una variante específica, basada en la pregunta: ¿ y^*_i satisface las restricciones de almacenamiento?

Si y^*_i satisface las restricciones de almacenamiento, el programa determina que y^*_i es óptimo y será la cantidad económica de pedido.

Caso contrario el programa va a funcionar de la siguiente manera:

Paso 1. _ Cuando las restricciones no se cumplen, el programa utilizará la función de Lagrange para calcular de manera óptima los valores restringidos de las cantidades de pedido, la función se estable de la siguiente manera:

$$\boldsymbol{L(\lambda, y_1, y_2, \ldots . y_n) = \sum_{i=1}^{n} \left(\frac{K_i D_i}{y_i} + \frac{h_i y_i}{2}\right) - \lambda\left(\sum_{i=1}^{n} a_i, y_i - A\right)} \qquad (15)$$

Paso 2. _ El siguiente paso una vez determinada la función de Lagrange es determinar los óptimos valores de λ y y_i, y para esto el programa analiza la siguiente formula:

$$\boldsymbol{y^*_i = \sqrt{\frac{2K_i D_i}{h_i - 2\lambda^* a_i}}} \qquad (16)$$

Si y^*_i aún no cumple con las restricciones, el programa procederá al paso 3. De lo contrario, determinará que la cantidad económica de pedido es y^*_i.

Paso 3. _ Para continuar con los cálculos necesarios, el programa comienza a decrementar λ en pequeñas cantidades. Este proceso se repetirá tantas veces como sea necesario hasta que y^*_i cumpla con las restricciones. Una vez concluidos todos estos cálculos, el programa dará las cantidades optimas de pedido para cada artículo según las restricciones del almacén.

4.1.4.2.4. Flujograma modelo sin costo de preparación

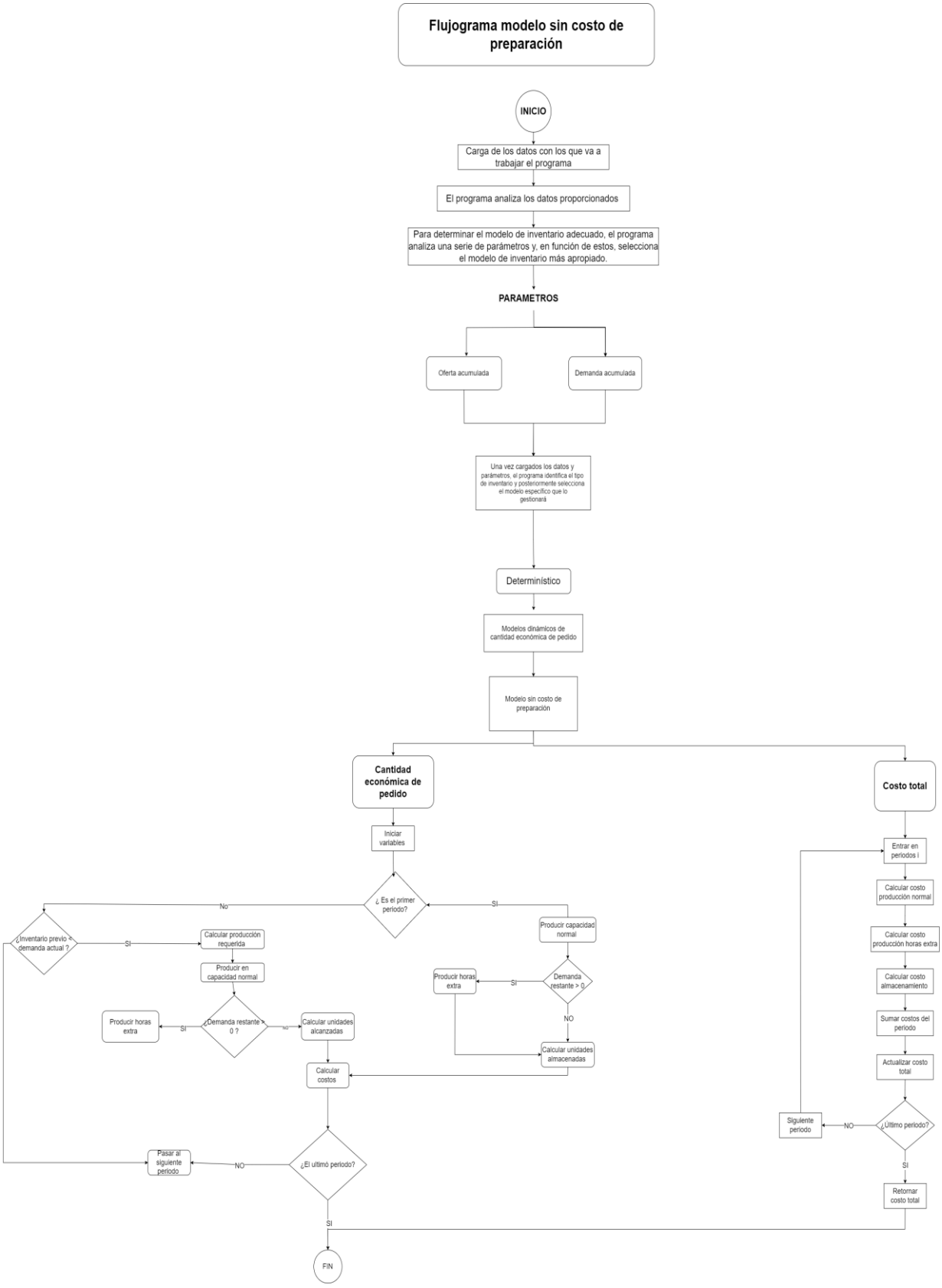


Figura 7. Flujograma modelo sin costo de preparación

En la Figura 7 se presenta el modelo sin costo de preparación, donde se analiza el parámetro de la oferta y demanda acumulada. Tras examinar la información, se identifica el tipo de modelo más apropiado para la simulación.

Este modelo sirve como una gran ayuda a la ahora de ser una herramienta la gestión de inventarios para optimizar la producción y el almacenamiento a lo largo de un horizonte de tiempo con múltiples periodos también se caracteriza por no incluir costos asociados a iniciar un lote de producción, lo que lo simplifica en comparación con otros modelos que sí los consideran.

Se caracteriza por minimizar los costos totales de producción y almacenamiento, garantizando que la demanda de cada periodo se satisfaga sin incurrir en faltantes. Mediante este enfoque, las empresas pueden planificar su producción de manera eficiente, aprovechando al máximo su capacidad y reduciendo el costo de mantener inventarios.

Paso 1. _ La demanda y la capacidad de producción son factores clave en la planificación de la producción. En cada periodo j, se cuenta con una demanda conocida, denotada como D_j, que representa la cantidad de productos requeridos.

No obstante, la capacidad de producción por periodo i es limitada, lo que implica restricciones en cuanto a cuántos productos pueden fabricarse en ese tiempo. La gestión eficiente de estos dos aspectos es crucial para equilibrar la oferta y la demanda, optimizando recursos y evitando sobrecostos o desabastecimientos.

Paso 2. _ El costo de producción es un periodo i es:

$$\boldsymbol{C_{i,j} = C_i + H_i \cdot (j - i)} \tag{17}$$

Paso 3. _ Las restricciones de producción establecen que la producción acumulada hasta el periodo i debe ser igual o superior a la demanda acumulada hasta ese punto.

$$\boldsymbol{\sum_{k=1}^{i} P_k \geq \sum_{k=1}^{i} D_k} \tag{18}$$

Paso 4. _ Para resolver el modelo, cada periodo se interpreta como una fuente (capacidad de producción) y un destino (demanda). El costo de "transporte" desde

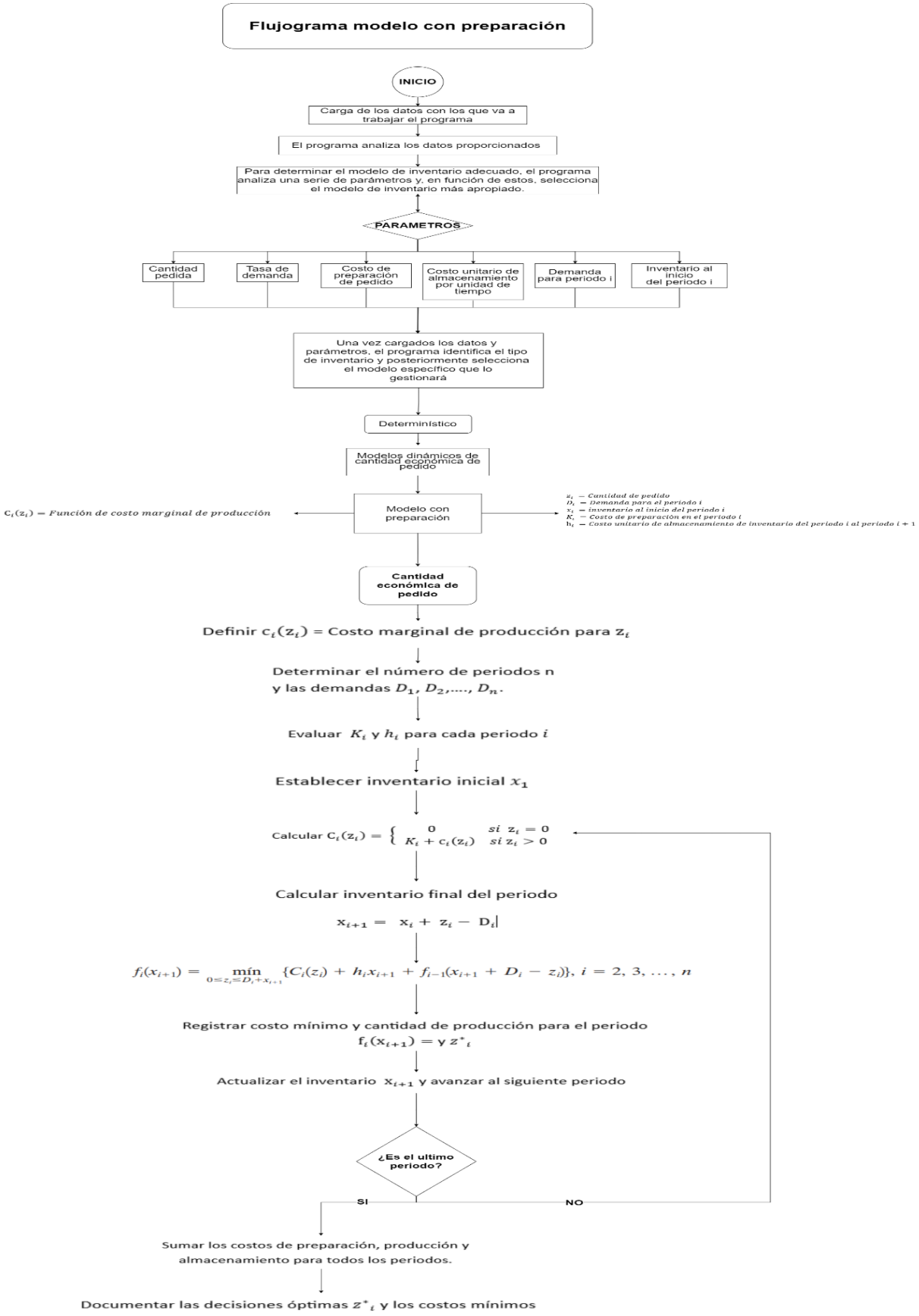
un origen i a un punto de llegada j incluye tanto los costos de producción como los de almacenamiento, en caso de ser necesario.

Paso 5. _ El resultado óptimo se encuentra asignando cantidades de producción en los periodos con costos más bajos, mientras se minimizan los costos de almacenamiento.

$$\mathbf{min} \sum_{i=1}^{n} \sum_{j=i}^{n} C_{i,j} \cdot C_{i,j} \tag{19}$$

Este modelo tiene un enfoque asegura que la demanda sea satisfecha en cada periodo sin faltantes, haciendo más eficiente la gestión de recursos. Este modelo es ideal para empresas que buscan reducir costos operativos en entornos de producción dinámica.

4.1.4.2.5. Flujograma modelo con preparación


Flujograma modelo con preparación
INICIO
Carga de los datos con los que va a trabajar el programa
El programa analiza los datos proporcionados
Para determinar el modelo de inventario adecuado, el programa analiza una serie de parámetros y, en función de estos, selecciona el modelo de inventario más apropiado.
PARAMETROS
Cantidad pedida
Tasa de demanda
Costo de preparación de pedido
Costo unitario de almacenamiento por unidad de tiempo
Demanda para periodo i
Inventario al inicio del periodo i
Una vez cargados los datos y parámetros, el programa identifica el tipo de inventario y posteriormente selecciona el modelo específico que lo gestionará
Determinístico
Modelos dinámicos de cantidad económica de pedido
Modelo con preparación
$c_i(z_i) = Función\ de\ costo\ marginal\ de\ producción$
z_i – Cantidad de pedido
D_i – Demanda para el periodo i
x_i – inventario al inicio del periodo i
K_i – Costo de preparación en el periodo i
h_i – Costo unitario de almacenamiento de inventario del periodo i al periodo i + 1
Cantidad económica de pedido
Definir $c_i(z_i)$ = Costo marginal de producción para z_i
Determinar el número de periodos n y las demandas $D_1, D_2, \ldots, D_n$.
Evaluar K_i y h_i para cada periodo i
Establecer inventario inicial x_1
Calcular $C_i(z_i) = \begin{cases} 0 & si\ z_i = 0 \\ K_i + c_i(z_i) & si\ z_i > 0 \end{cases}$
Calcular inventario final del periodo
$x_{i+1} = x_i + z_i - D_i$
$f_i(x_{i+1}) = \min_{0 \le z_i \le D_i + x_{i+1}} \{C_i(z_i) + h_i x_{i+1} + f_{i-1}(x_{i+1} + D_i - z_i)\},\ i = 2, 3, \ldots, n$
Registrar costo mínimo y cantidad de producción para el periodo
$f_i(x_{i+1})$ = y z^*_i
Actualizar el inventario x_{i+1} y avanzar al siguiente periodo
¿Es el ultimo periodo?
SI
NO
Sumar los costos de preparación, producción y almacenamiento para todos los periodos.
Documentar las decisiones óptimas z^*_i y los costos mínimos

En la Figura 8, se ilustra la lógica que sigue el modelo con preparación. La figura detalla todos los parámetros y la secuencia de procesos que este modelo sigue dentro del programa.

Para que este modelo funcione correctamente, es esencial cargar los datos con los siguientes parámetros específicos:

$\boldsymbol{Z_i}$ **=**Cantidad de pedido

$\boldsymbol{D_i}$ **=**Demanda para el periodo i

$\boldsymbol{X_i}$ **=**inventario al inicio del periodo i

$\boldsymbol{K_i}$ **=**Costo de preparación en el periodo i

$\boldsymbol{h_i}$ **=**Costo unitario de almacenamiento de inventario del periodo i al periodo i+1

El programa identifica estos parámetros e inmediatamente se desvía hacia el modelo con preparación. Una vez determinado que este es el modelo de inventario adecuado, el programa sigue una serie de pasos específicos para llevar a cabo los cálculos necesarios. Estos pasos aseguran que el proceso se realice de manera precisa y eficiente. A continuación, se describen los pasos que sigue el programa para este modelo en particular:

Paso 1. _ El primer paso que sigue el programa para la resolución de este modelo, es definir la función de costo marginal de producción para Z_i

Paso 2. _ Una vez definido esta función el programa establece el número de periodos n y las demandas $D_1, D_2, \ldots, D_n$

Paso 3. _ Para empezar con los cálculos de cada periodo el programa evalúa K_i y h_i por cada periodo que tienen los datos

Paso 4. _ El programa establece el inventario inicial X_1

Paso 5. _ El programa calcula la función de costos que es la siguiente:

$$\boldsymbol{C_i(Z_i)} = \begin{cases} \boldsymbol{0} & \boldsymbol{si\ Z_i = 0} \\ \boldsymbol{K_i + C_i(Z_i)} & \boldsymbol{si\ Z_i > 0} \end{cases} \quad (20)$$

Paso 6. _ Después de esto el programa calcula el inventario final del periodo, con la siguiente formula:

$$\boldsymbol{x_{i-1} = x_i + Z_i - D_i} \quad (21)$$

Y una vez determinado todos estos datos se puede hacer uso de fórmula para calcular el costo mínimo del inventario para los períodos correspondientes.

$$f_i(x_{i+1}) = \min_{0 \le Z_i \le D_i + x_{i+1}} \{ C_i(Z_i) + h_i x_{i+1} + f_{i-1}(x_{i+1} + D_i - Z_i) \}, i = 2, 3, \dots, n \quad (22)$$

Paso 7. _ Con esta fórmula que el programa lo calcula, el siguiente paso que realiza el programa es actualizar el inventario x_{i+1}, para avanzar al siguiente periodo

A partir de este punto, el programa puede tomar una de dos decisiones según una variante específica, basada en la pregunta: $¿Es\ el\ ultimo\ periodo?$

En caso de ser el último periodo:

Paso 8. _ El programa calcula la suma de los costos de preparación y almacenamiento para cada período, y luego obtiene las decisiones óptimas de Z^*_i y los costos más bajos.

Y en caso de no ser el último periodo:

Paso 1. _ En ese caso el programa con los nuevos datos que ya calculo repite todos los procesos y pasos desde el paso 5, hasta llegar al último periodo y determinar la cantidad optimas de pedido

4.1.4.3. Modelos probabilísticos

En esta sección se evidencia la elaboración y explicación detallada de los flujogramas de únicamente los modelos probabilísticos, de igual manera que en el anterior punto, resulta fundamental señalar que estos flujogramas han sido desarrollados tomando como referencia a Taha (2004).

4.1.4.3.1. Flujograma modelo "probabilizado" de cantidad económica de pedido

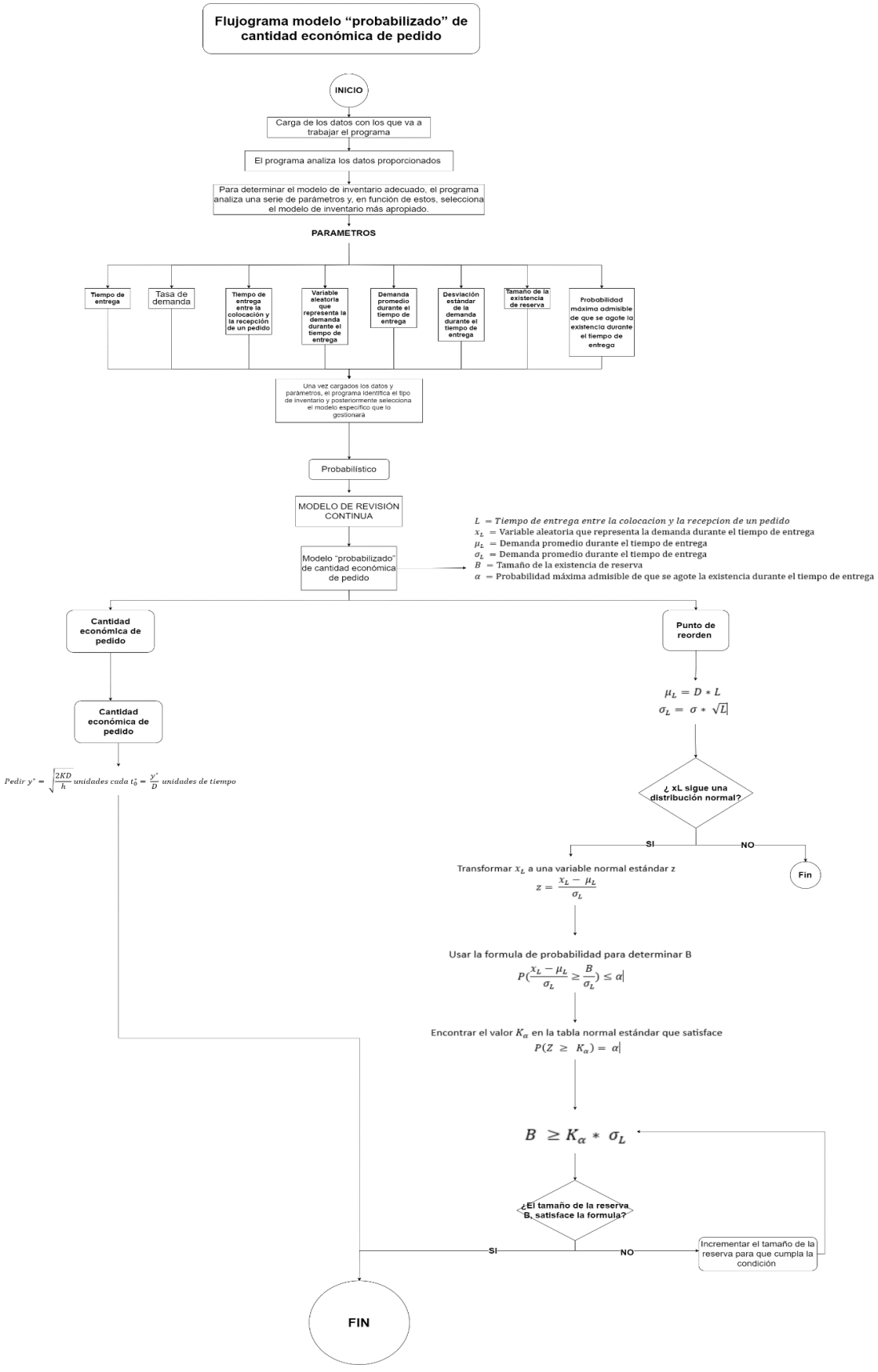


Figura 9. Flujograma modelo "probabilizado" de cantidad económica de pedido

En la Figura 9, se ilustra la lógica que sigue el modelo "probabilizado" de cantidad económica de pedido. La figura detalla todos los parámetros y la secuencia de procesos que este modelo sigue dentro del programa.

Para que este modelo funcione correctamente, es esencial cargar los datos con los siguientes parámetros específicos:

D = Tasa de demanda (u/u de tiempo)

K = Costo de preparación correspondiente a la colocación de un pedido ($/pedido)

h = Costo de almacenamiento ($ por unidad en inventario por unidad de tiempo)

L = Tiempo de entrega entre la colocación y la recepción de un pedido

$\boldsymbol{x_L}$ **=** Variable aleatoria que representa la demanda durante el tiempo de entrega
$\boldsymbol{\mu_L}$ **=** Demanda promedio durante el tiempo de entrega
$\boldsymbol{\sigma_L}$ **=** Desviación estándar de la demanda durante el tiempo de entrega

B = Tamaño de la existencia de reserva
$\boldsymbol{\alpha}$ **=** Probabilidad máxima admisible de que se agote la existencia durante el tiempo de entrega

El programa identifica estos parámetros e inmediatamente se desvía hacia el modelo "probabilizado" de cantidad económica de pedido. Una vez determinado que este es el modelo de inventario adecuado, el programa sigue una serie de pasos específicos para llevar a cabo los cálculos necesarios. Estos pasos aseguran que el proceso se realice de manera precisa y eficiente. A continuación, se describen los pasos que sigue el programa para este modelo en particular:

Paso 1. _ para el cálculo

de la cantidad económica de pedido, el programa usa los parámetros K, D y h. Cuando el programa identifica estos parámetros hace uso de la formula:

$$\boldsymbol{y^*} = \sqrt{\frac{\boldsymbol{2KD}}{\boldsymbol{h}}} \qquad (6)$$

Esta fórmula permite al programa determinar el lote óptimo de pedido y el nivel de reabastecimiento para minimizar los costos totales de inventario.

Paso 2. _ Una vez determinada la cantidad a pedir, el programa hace el cálculo de en qué tiempo se deben pedir estas cantidades de pedido, para esto hace uso de los parámetros D y la reciente cantidad de pedido calculada y^*, en la formula:

$$\boldsymbol{t^*_o = \frac{y^*}{D}} \tag{7}$$

Esta fórmula permite al programa determinar el intervalo de tiempo ideal entre pedidos. Al aplicar esta fórmula, el programa optimiza el momento para realizar nuevos pedidos, asegurando una gestión eficiente del inventario y minimizando tanto los costos de almacenamiento como la posibilidad de agotar existencias.

Paso 3. _ Para calcular punto de reorden, programa hace uso de las dos siguientes formulas:

$$\boldsymbol{\mu_L = DL} \tag{23}$$

$$\boldsymbol{\sigma_L = \sqrt{\sigma^2 L}} \tag{24}$$

En la formula σ_L el valor de L debe redondearse a un entero

A partir de este punto, el programa puede tomar una de dos decisiones según una variante específica, basada en la pregunta: ¿x_L sigue una distribución normal?

En caso x_L no siga una distribución normal, el programa no puede continuar con los cálculos y para toda operación.

En caso x_L siga una distribución normal, el programa continúa con los siguientes procesos:

Paso 4. _ En este punto es necesario transformar x_L en una variable estándar. Para ello, el programa utiliza y calcula la siguiente fórmula:

$$\boldsymbol{z = \frac{x_L - \mu_L}{\sigma_L}} \tag{25}$$

Paso 5. _ El programa necesita determinar B, para continuar con los cálculos, entonces para hacer el cálculo con la formula:

$$\boldsymbol{P}\left(\frac{\boldsymbol{x_L - \mu_L}}{\boldsymbol{\sigma_L}} \geq \frac{\boldsymbol{B}}{\boldsymbol{\sigma_L}}\right) \leq \boldsymbol{\alpha} \quad (26)$$

Paso 6. _ Para seguir con el proceso, también se debe identificar y encontrar la variable K_α en la tabla de distribución estándar que satisface la siguiente condición.

$$\boldsymbol{P(Z \geq K_\alpha) = \alpha} \quad (27)$$

Paso 7. _ Para calcular el tamaño de la reserva el programa procede a usar la formula:

$$\boldsymbol{B \geq \sigma_L * K_\alpha} \quad (28)$$

A partir de este punto, el programa puede tomar una de dos decisiones según una variante específica, basada en la pregunta: ¿El tamaño de la reserva B, satisface la formula?

Si no satisface la formula entonces el programa incrementa el tamaño de la reserva hasta que cumpla la condición.

Si satisface la fórmula:

Paso 8. _ Se calcula el punto de reorden con la siguiente fórmula:

$$\boldsymbol{Punto\ reorden = B + \mu_L} \quad (29)$$

4.1.4.3.2. Flujograma modelo probabilístico de cantidad económica de pedido

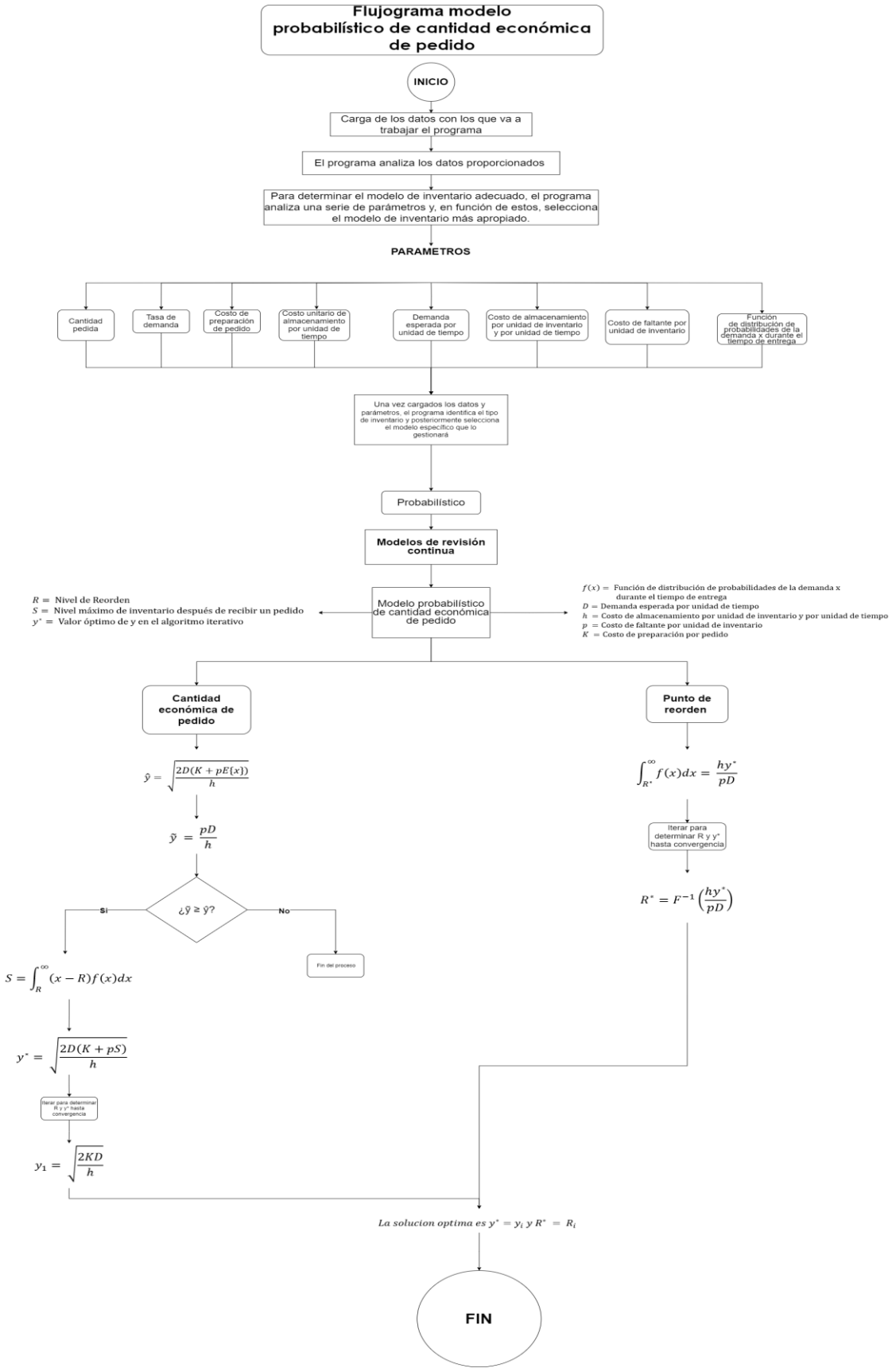


Figura 10. Flujograma modelo probabilístico de cantidad económica de pedido

En la Figura 10, se ilustra la lógica que sigue el modelo probabilístico de cantidad económica de pedido. La figura detalla todos los parámetros y la secuencia de procesos que este modelo sigue dentro del programa.

Para que este modelo funcione correctamente, es esencial cargar los datos con los siguientes parámetros específicos:

f(x)= función de distribución de probabilidades de la demanda x durante el

tiempo de entrega

D = Demanda esperada por unidad de tiempo

h = Costo de almacenamiento por unidad de inventario y por unidad de tiempo

p = Costo de faltante por unidad de inventario

K = Costo de preparación por pedido

El programa identifica estos parámetros e inmediatamente se desvía hacia el modelo probabilístico de cantidad económica de pedido. Una vez determinado que este es el modelo de inventario adecuado, el programa sigue una serie de pasos específicos para llevar a cabo los cálculos necesarios. Estos pasos aseguran que el proceso se realice de manera precisa y eficiente. A continuación, se describen los pasos que sigue el programa para este modelo en particular:

Paso 1. _ El primer paso que va a realizar el sistema es comprobar que tenga una solución que cumpla las condiciones, para esto usa y calcula las dos siguientes formulas:

$$\hat{y} = \sqrt{\frac{\boldsymbol{2D(K+pE\{X\})}}{\boldsymbol{h}}} \tag{30}$$

$$\tilde{\boldsymbol{y}} = \frac{\boldsymbol{pD}}{\boldsymbol{h}} \tag{31}$$

A partir de este punto, el programa puede tomar una de dos decisiones según una variante específica, basada en la pregunta: ¿ $\tilde{y} \geq \hat{y}$?

Si $\tilde{y} < \hat{y}$, entonces el programa finaliza el proceso, ya que al no cumplirse esta condición se determina que no existen valores óptimos para los datos proporcionados.

Si se cumple que $\tilde{y} \geq \hat{y}$, entonces el programa continúa de la siguiente manera:

Paso 2. _ Como se cumplió la condición el programa determina que hay soluciones únicas para y^* y R^*, y para calcular estas variables el programa calcula S con la siguiente integral:

$$\boldsymbol{S = \int_{R}^{\infty} (x - R) f(x) dx} \tag{32}$$

Paso 3. _ El programa sigue con el proceso y ahora con la fórmula para calcular la cantidad optima de pedido que es:

$$\boldsymbol{y^* = \sqrt{\frac{2D(K + pS)}{h}}} \tag{33}$$

$$\boldsymbol{\int_{R}^{\infty} f(x) dx = \frac{h y^*}{pD}} \tag{34}$$

Paso 4. _ Una vez el programa calcula estas fórmulas y usa S en ambas funciones, procede a usar la siguiente fórmula para poder seguir con el proceso.

$$\boldsymbol{R_i = K - \frac{y_i}{E\{x\}}} \tag{35}$$

Paso 5. _ Con las fórmulas ya calculadas, el programa realiza iteraciones para determinar R y y^* hasta que tengan convergencia, y una vez el programa logra esto da los valores obtenidos de cantidad a pedir y^* y el puno de reorden R.

4.1.4.3.3. Flujograma modelo sin preparación

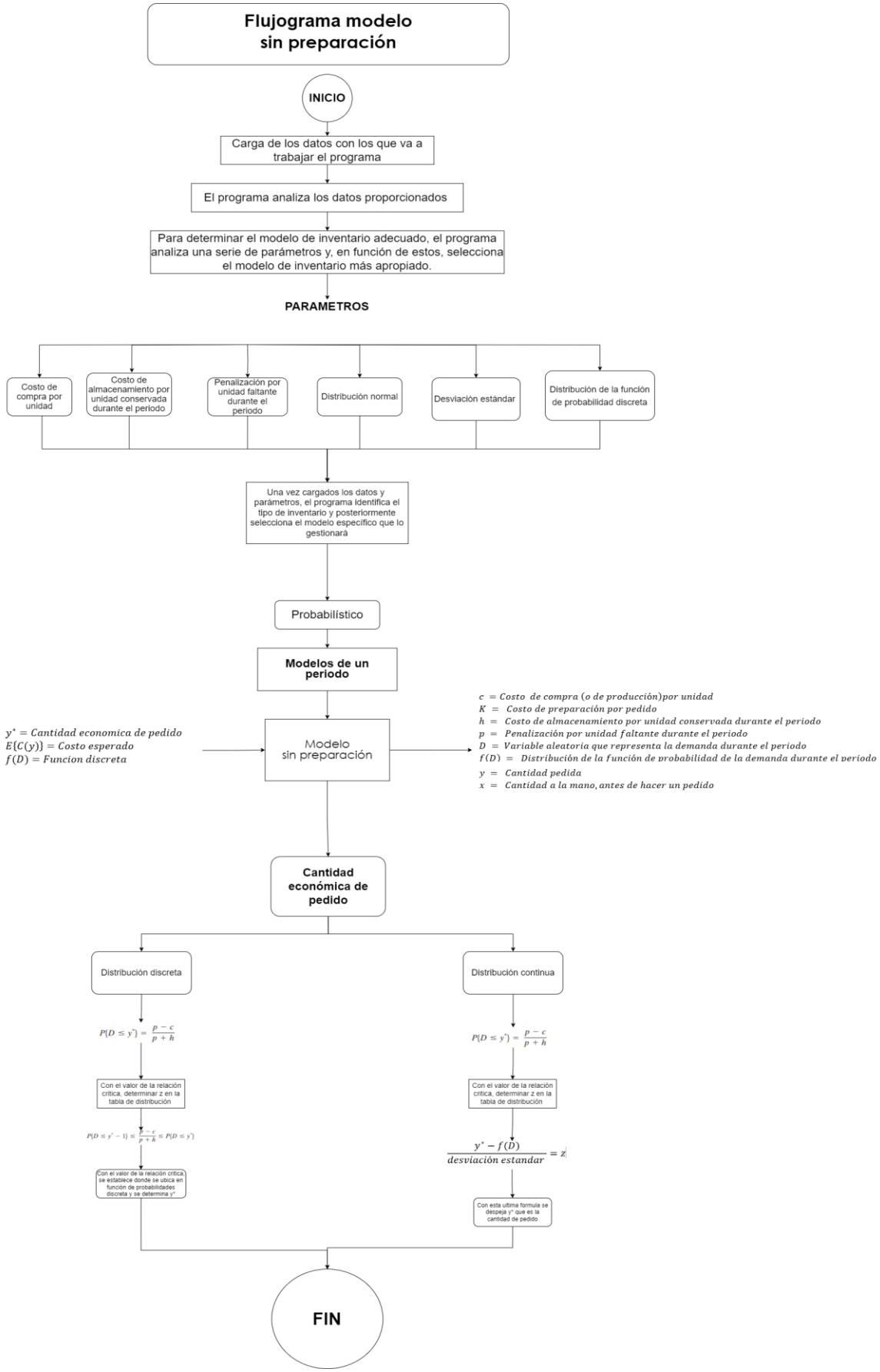


Figura 11. Flujograma modelo sin preparación

En la Figura 11, se ilustra la lógica que sigue el modelo probabilístico de cantidad económica de pedido. La figura detalla todos los parámetros y la secuencia de procesos que este modelo sigue dentro del programa.

Para que este modelo funcione correctamente, es esencial cargar los datos con los siguientes parámetros específicos:

c = Costo de compra (o de producción) por unidad

h = Costo de almacenamiento por unidad conservada durante el periodo

p = Penalización por unidad faltante durante el periodo

D = Variable aleatoria que representa la demanda durante el periodo

f(D) = Distribución de la función de probabilidad de la demanda durante

el periodo

y = Cantidad pedida

x = Cantidad a la mano, antes de hacer un pedido

El programa identifica estos parámetros e inmediatamente se desvía hacia el modelo sin preparación. Una vez determinado que este es el modelo de inventario adecuado, el programa sigue una serie de pasos específicos para llevar a cabo los cálculos necesarios. Estos pasos aseguran que el proceso se realice de manera precisa y eficiente. A continuación, se describen los pasos que sigue el programa para este modelo en particular:

Distribución continua

Paso 1. _ Como primer paso, se debe calcular la relación crítica con esta fórmula, este valor debe ser positivo.

$$\frac{\boldsymbol{p-c}}{\boldsymbol{p+h}} \tag{36}$$

Paso 2. _ Una vez determinada la relación critica, el programa toma este valor para calcularlo en la tabla de distribución el valor z.

Paso 3. _ Con el valor de z definido por la tabla de distribución, se hace uso de la siguiente formula:

$$\frac{\boldsymbol{y^* - f(D)}}{\boldsymbol{desviación\ estandar}} = \boldsymbol{z} \tag{37}$$

Paso 4. _ Con esta fórmula se despeja y*, y esa es la cantidad para pedir.

Distribución discreta:

Paso 1. _ Como primer paso, se debe calcular la relación critica con esta fórmula, este valor debe ser positivo.

$$\frac{\boldsymbol{p - c}}{\boldsymbol{p + h}} \tag{36}$$

Paso 2. _ Con el valor cálculo de la relación critica el programa hace uso de la siguiente condición:

$$\boldsymbol{P\{D \leq y^* - 1\} \leq \frac{p - c}{p + h} \leq P\{D \leq y^*\}} \tag{38}$$

Con esta condición el programa determina de entre la función de probabilidad discreta, donde se encuentra la relación crítica y ese es el valor para pedir.

4.1.4.3.4. Flujograma modelo con preparación (política s-S)

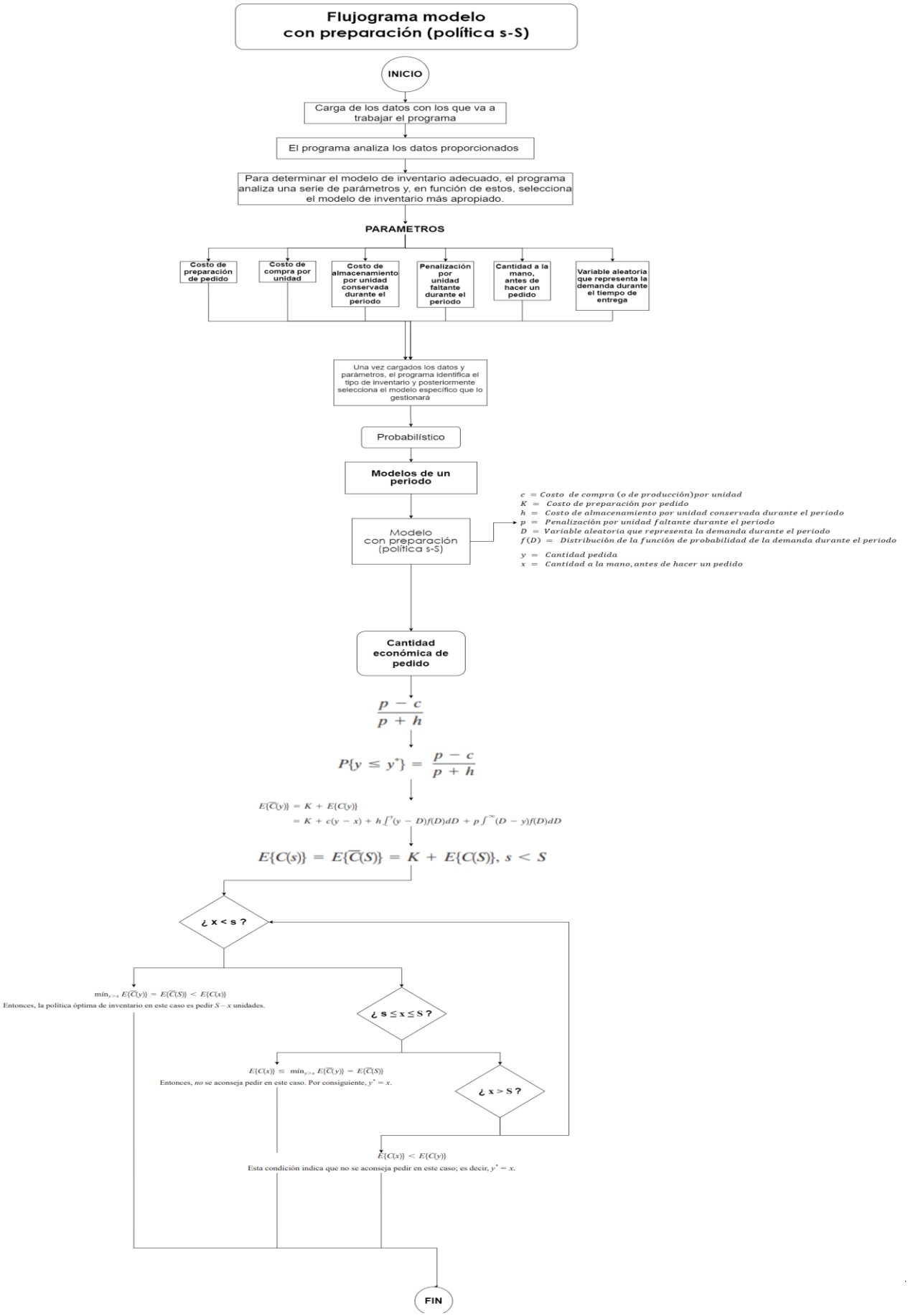


Figura 12. Flujograma modelo con preparación (política s-S)

En la Figura 12, se ilustra la lógica que sigue el modelo con preparación (política s-S). La figura detalla todos los parámetros y la secuencia de procesos que este modelo sigue dentro del programa.

Para que este modelo funcione correctamente, es esencial cargar los datos con los siguientes parámetros específicos:

c = Costo de compra (o de producción) por unidad

K = Costo de preparación por pedido

h = Costo de almacenamiento por unidad conservada durante el periodo

p = Penalización por unidad faltante durante el periodo

D = Variable aleatoria que representa la demanda durante el periodo

f(D) = Distribución de la función de probabilidad de la demanda durante

el periodo

El programa identifica estos parámetros e inmediatamente se desvía hacia el modelo con preparación (política s-S). Una vez determinado que este es el modelo de inventario adecuado, el programa sigue una serie de pasos específicos para llevar a cabo los cálculos necesarios. Estos pasos aseguran que el proceso se realice de manera precisa y eficiente. A continuación, se describen los pasos que sigue el programa para este modelo en particular:

Paso 1. _ El primer paso que realiza el programa para este modelo de inventario es determinar la relación critica con la formula:

Paso 2. _ Una vez calculada la relación critica, para calcular la cantidad de pedido y^* se necesita tener los límites de la función de distribución de probabilidades de la demanda uniforme, se hace de la formula:

$$\boldsymbol{P\{D \leq y^*\} = \int_0^{y^*} \frac{limite\ inferior}{limite\ superior} dD} \tag{39}$$

Paso 3. _ Cuando el programa calcula los valores de estas dos fórmulas, entonces puede determinar el valor de S, y una vez calculado S, el programa sigue con los procesos con la función de costos esperados por periodo.

$$\boldsymbol{E\{\overline{C}(y)\} = K + E\{C(y)\}} \tag{40}$$

$$E\{\overline{C}(y)\} = K + c(y - x) + h\int_0^y (y - D)\, f(D)dD + p\int_y^\infty (D - y)\, f(D)dD \quad (41)$$

Paso 4. _ Esta ecuación da como resultado una ecuación cuadrática con la variable y, entonces en este próximo paso el programa cambia de variable a s, con la ecuación.

$$E\{C(y)\} = K + E\{C(S)\},\ s < S \quad (42)$$

Paso 5. _ Una vez calculado esta ecuación, el programa ya determino los valores de s, S, x. Y una vez calculados estos valores el programa actúa dependiendo de los valores.

A partir de este punto, el programa puede tomar una de 3 decisiones según una variante específica, basada en las preguntas:

- **¿x < s?, de ser este el caso:** La política de inventario debe ser pedir S-x unidades
- **¿s ≤ x ≤ S?, de ser este el caso:** No es prudente hacer un pedido, entonces la política de inventario es $y^* = x$
- **¿x > S? de ser este el caso:** No es prudente hacer un pedido, entonces la política de inventario es $y^* = x$

4.1.4.3.5. Flujograma modelo de varios periodos

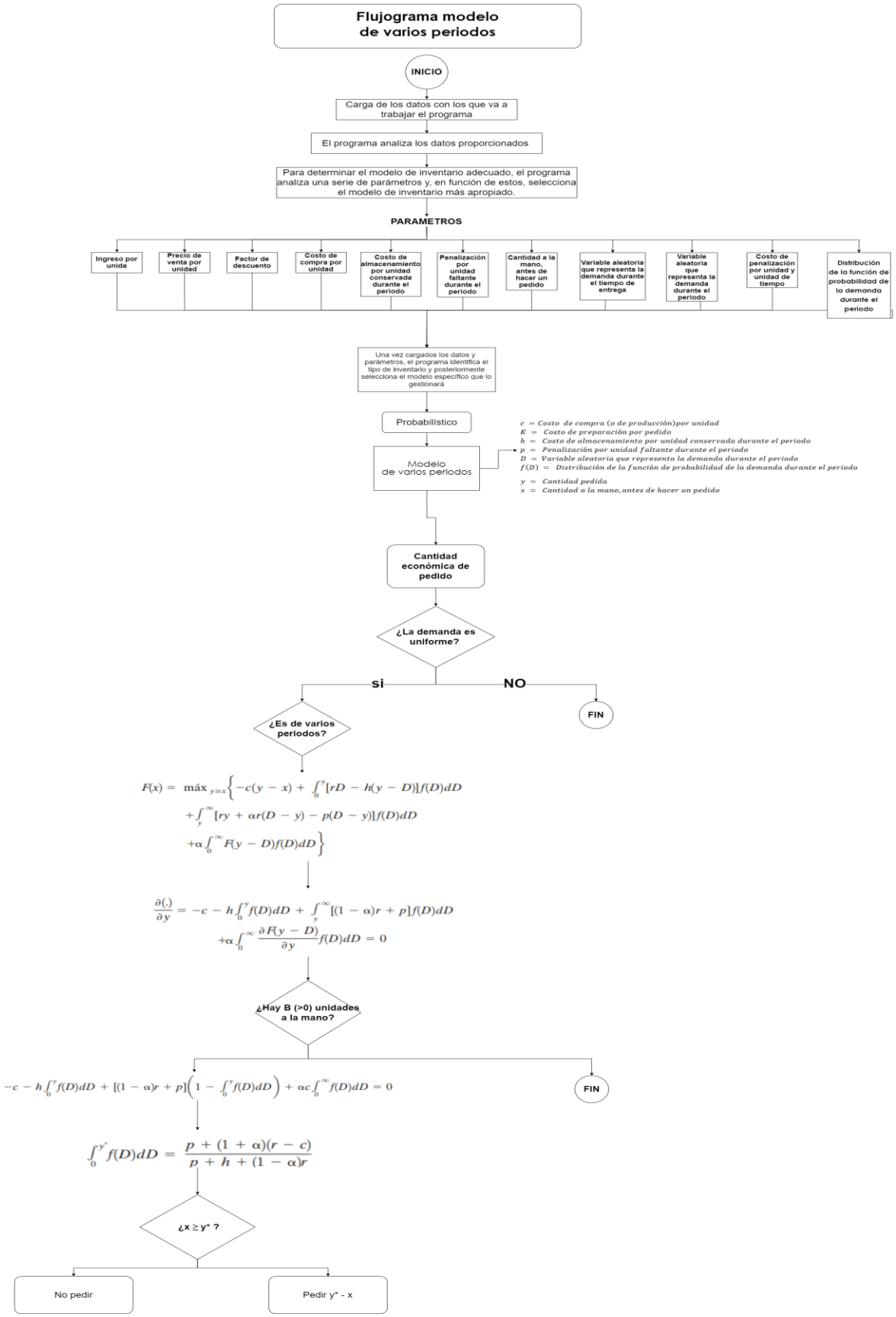


Figura 13. flujograma modelo de varios periodos

En la Figura 13, se ilustra la lógica que sigue el modelo de varios periodos La figura detalla todos los parámetros y la secuencia de procesos que este modelo sigue dentro del programa.

Para que este modelo funcione correctamente, es esencial cargar los datos con los siguientes parámetros específicos:

c = Costo de compra (o de producción) por unidad

K = Costo de preparación por pedido

h = Costo de almacenamiento por unidad conservada durante el periodo

p = Penalización por unidad faltante durante el periodo

D = Variable aleatoria que representa la demanda durante el periodo

f(D) = Distribución de la función de probabilidad de la demanda durante

el periodo

y = Cantidad pedida

x = Cantidad a la mano, antes de hacer un pedido

r = ingreso por unidad

El programa identifica estos parámetros e inmediatamente se desvía hacia el modelo de varios periodos. Una vez determinado que este es el modelo de inventario adecuado, el programa sigue una serie de pasos específicos para llevar a cabo los cálculos necesarios. Estos pasos aseguran que el proceso se realice de manera precisa y eficiente. A continuación, se describen los pasos que sigue el programa para este modelo en particular:

Paso 1. _ Primero, el programa verifica si la demanda es uniforme. Si se determina que la demanda es uniforme, el programa continúa con los siguientes pasos. En caso contrario, detiene su proceso y finaliza la operación.

Paso 2. _ Una vez que el programa verifica que la demanda es uniforme, procede a determinar si los datos abarcan varios períodos. Si se confirma que los datos comprenden múltiples períodos, el programa continúa con el proceso. En caso contrario, finaliza la operación y detiene todas las actividades adicionales.

Paso 3. _ Una vez verificado todos estos pasos el programa empieza con el modelo de programación dinámica:

$$F_i(x_i) = \max_{y_i \geq x_i}\{ -c(y_i - x_i) + \int_0^y [rD - h(y_i - D)]f(D)dD + \int_{y_i}^{\infty} [ry_i + \propto r(D - y_i) - p(D - y_i)]f(D)dD + \propto \int_0^{\infty} F_{i+1}(y_i - D)f(D)dD \}, i = 1, 2, \ldots, n \quad (43)$$

Paso 4. _ Para determinar el valor de y, el programa hace uso de la siguiente condición:

$$\frac{\partial(.)}{\partial y} = -c - h\int_0^y f(D)dD + \int_y^{\infty} [(1-\propto)r + p]f(D)dD + \propto \int_0^{\infty} \frac{\partial F(y-D)}{\partial y} f(D)dD = 0 \quad (44)$$

A partir de este punto, el programa puede tomar una de dos decisiones según una variante específica, basada en la pregunta: ¿Hay β (>0) unidades a la mano al iniciar el periodo?

Si no se cumple la condición, el programa finaliza su proceso de inmediato, deteniendo todas las operaciones. Sin embargo, si la condición se cumple, el programa procede al siguiente conjunto de procesos y cálculos necesarios para la gestión eficiente del inventario.

Paso 5. _ Para determinar la política optima de inventario de cada inventario el programa utiliza:

$$\int_0^{y^*} f(D)dD = \frac{P + (1+\propto)(r - c)}{p + h + (1-\propto)r} \quad (45)$$

A partir de este punto, el programa puede tomar una de dos decisiones según una variante específica, basada en la pregunta: ¿ x ≥ y^* ?

SI x ≥ y^* :

El programa reconoce esta condición y determina que en este caso no se debe realizar el pedido

Por otro lado, si x < y*:

El programa reconoce esta condición y determina que en este caso se debe pedir y^*- x.

Con este último modelo, se han cubierto exhaustivamente todos los tipos de inventarios y las funciones que el programa realiza. Inicialmente, la intención era enfocarse únicamente en los modelos de inventarios más utilizados en Tulcán. Sin embargo, en el transcurso del desarrollo del programa, se decidió ampliar el alcance para incluir todos los modelos de inventarios. Esta expansión permite que el programa no solo sea más robusto y versátil, sino que también tenga un mayor potencial de aplicación en diversas situaciones y contextos empresariales.

Este enfoque integral no solo asegura que el programa pueda adaptarse a diferentes necesidades y circunstancias, sino que también incrementa su valor como herramienta de gestión de inventarios. Al incluir una gama completa de modelos, se ha fortalecido la capacidad del programa para proporcionar soluciones óptimas y eficaces, independientemente de la complejidad o especificidad de los requerimientos de inventario.

La inclusión de todos los modelos de inventarios, desde los más básicos hasta los más avanzados y especializados, le da al programa una profundidad y crecimiento significativos. Esta versatilidad permite que los usuarios se beneficien de una herramienta integral que puede abordar cualquier desafío relacionado con la gestión de inventarios, facilitando la toma de decisiones informadas y estratégicas.

4.1.4.4. Códigos del programa

Una vez que se definió la estructura del programa tanto de manera general como específica para los diferentes tipos de modelos de inventarios mediante flujogramas detallados, se procedió con la implementación práctica de estos diseños. De acuerdo con el objetivo 3 establecido, el lenguaje de programación seleccionado para desarrollar el sistema fue *Python*. Durante esta etapa, se tradujo toda la lógica delineada en los flujogramas a código *Python*, lo cual permitió estructurar el programa de manera efectiva.

Inicialmente, se desarrollaron códigos individuales para cada modelo de inventario, lo que implicó la creación de funciones y algoritmos específicos para abordar las

particularidades de cada uno. Posteriormente, estos códigos fueron integrados en dos programas separados: uno para los modelos determinísticos y otro para los modelos probabilísticos. Esta integración permitió que el programa fuera capaz de gestionar de manera eficiente todos los modelos de inventario definidos, siguiendo los pasos y procesos preestablecidos en los flujogramas correspondientes. Sin embargo, estos dos programas no representan la versión final de este trabajo.

Aunque ambos programas proporcionan los cálculos precisos para cada modelo específico de inventario, el objetivo principal de este proyecto va más allá. No se trata simplemente de calcular las soluciones para modelos de inventario de manera aislada, sino de implementar un enfoque basado en árboles de decisión, es decir, incorporar inteligencia artificial al programa para que pueda tomar decisiones de manera autónoma y adaptativa. La idea es utilizar estos dos programas como base para realizar simulaciones y generar un conjunto de datos que luego se integrará en el sistema con árboles de decisión. Este sistema, a través del aprendizaje de patrones a partir de la base de datos generada, será capaz de optimizar el manejo de inventarios de manera más inteligente, tomando decisiones informadas y adaptativas basadas en la información previamente recopilada.

Todos los códigos de los modelos de inventarios individuales Anexo 5 y de los códigos unificados por modelos están documentados en el Anexo 6 y Anexo 7, donde se detallan específicamente debido a la extensión que representaría incluirlos todos en este documento principal.

4.1.4.5. Simulación con los códigos individuales

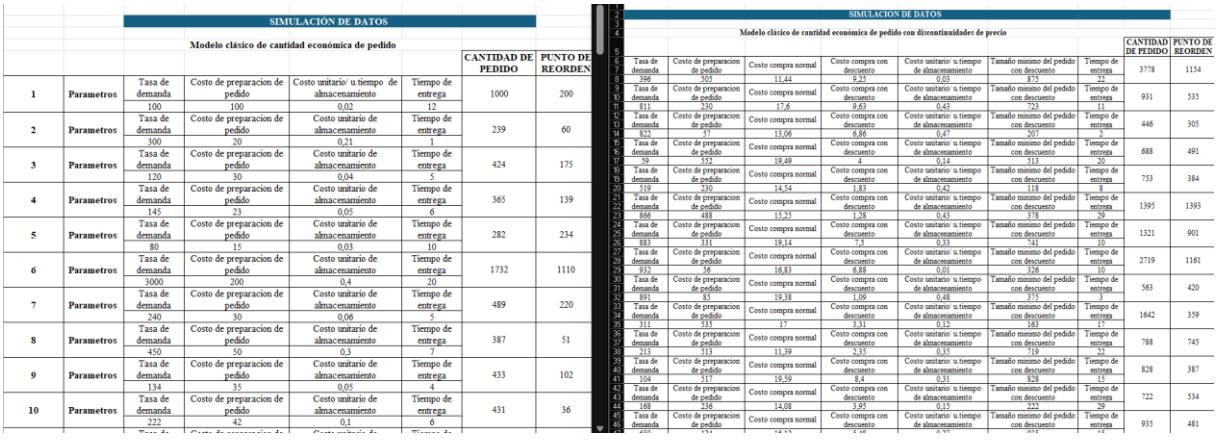
Se debe aclarar que esta parte es diferente al objetivo 5 de este trabajo, en este punto se utilizó los dos programas creados para realizar simulaciones de datos y crear la base que será usada en el programa principal. Para ello, se ejecutaron 100 simulaciones en todos los códigos y modelos de inventarios. Los resultados obtenidos de estas simulaciones fueron registrados en una hoja de *Excel*, proporcionando una base para el avance del proyecto. Estos datos permitieron la creación del programa final, en cuyo desarrollo surgió una mejora significativa en la metodología original.

Se optó por la implementación de inteligencia artificial. Este enfoque permitió la creación de un programa que únicamente requería como insumos las entradas y

salidas (parámetros y resultados) de cada modelo, sin necesidad de replicar el proceso completo de cálculo dentro del código.

La inteligencia artificial, a partir de los datos de las simulaciones realizadas con los códigos individuales, fue capaz de aprender y replicar los cálculos de manera automática. Así, el nuevo programa ya no necesitó la inclusión de los códigos originales en su estructura, sino solo los parámetros de entrada y los resultados generados. A pesar de ello, los códigos individuales fueron esenciales en la etapa de simulación, ya que proporcionaron la información necesaria para entrenar y optimizar el nuevo sistema basado en inteligencia artificial.

Para brindar un ejemplo representativo y de fácil acceso, en la Figura 14 y Figura 15 se ilustra cómo se registraron y documentaron las simulaciones de cada modelo, lo que permite observar la estructura utilizada para el almacenamiento de datos y los criterios empleados en la evaluación de los resultados. Este ejemplo es fundamental para comprender la metodología aplicada en el procesamiento de las simulaciones y en la interpretación de los datos obtenidos.



SIMULACIÓN DE DATOS

Modelo clásico de cantidad económica de pedido

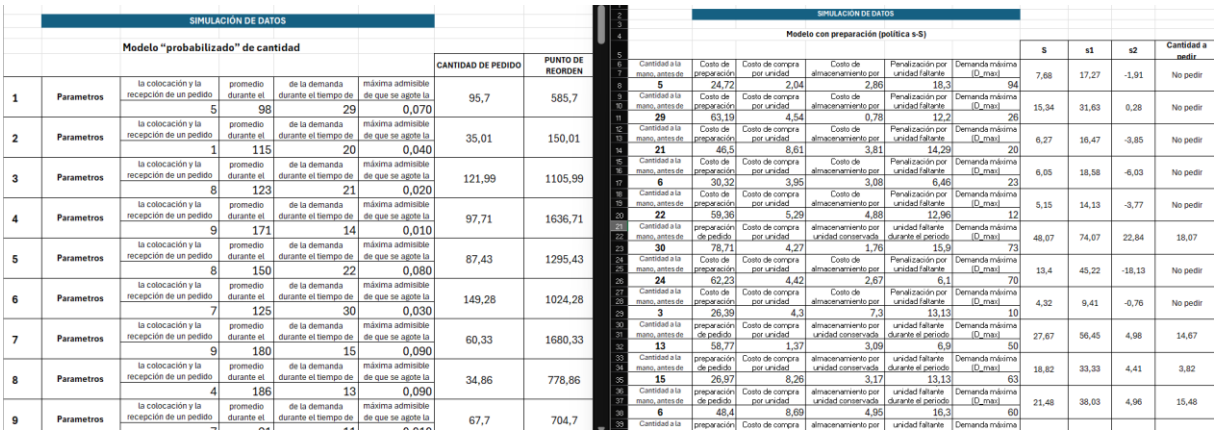
		Tasa de demanda	Costo de preparacion de pedido	Costo unitario/ u.tiempo de almacenamiento	Tiempo de entrega	CANTIDAD DE PEDIDO	PUNTO DE REORDEN
1	Parametros	100	100	0,02	12	1000	200
2	Parametros	300	20	0,21	1	239	60
3	Parametros	120	30	0,04	5	424	175
4	Parametros	145	23	0,05	6	365	139
5	Parametros	80	15	0,03	10	282	234
6	Parametros	3000	200	0,4	20	1732	1110
7	Parametros	240	30	0,06	5	489	220
8	Parametros	450	50	0,3	7	387	51
9	Parametros	134	35	0,05	4	433	102
10	Parametros	222	42	0,1	6	431	36

SIMULACION DE DATOS

Modelo clásico de cantidad económica de pedido con discontinuidades de precio

Tasa de demanda	Costo de preparacion de pedido	Costo compra normal	Costo compra con descuento	Costo unitario/ u.tiempo de almacenamiento	Tamaño mínimo del pedido con descuento	Tiempo de entrega	CANTIDAD DE PEDIDO	PUNTO DE REORDEN
396	505	11,44	9,25	0,03	875	22	3778	1154
811	230	17,6	9,63	0,43	725	11	931	535
822	57	13,06	6,86	0,47	207	2	446	305
59	552	19,49	4	0,14	513	20	688	491
519	230	14,54	1,81	0,42	118	8	753	384
166	488	15,25	1,28	0,43	378	29	1395	1393
883	331	19,14	7,5	0,33	741	10	1321	901
932	56	16,83	6,88	0,01	326	10	2719	1161
891	85	19,38	1,09	0,48	375	3	563	420
311	535	17	3,31	0,12	183	17	1642	359
213	513	11,39	3,35	0,35	719	22	788	745
104	517	19,59	8,4	0,31	828	15	828	387
168	236	14,08	3,95	0,15	222	29	722	534
222	42	0,1	6				935	481

Figura 14. Simulación de datos modelo *EOQ* y *EOQ* con discontinuidad de precios



SIMULACIÓN DE DATOS

Modelo "probabilizado" de cantidad

						CANTIDAD DE PEDIDO	PUNTO DE REORDEN
1	Parametros	la colocación y la recepción de un pedido	promedio durante el	de la demanda durante el tiempo de	máxima admisible de que se agote la	95,7	585,7
		5	98	29	0,070		
2	Parametros	la colocación y la recepción de un pedido	promedio durante el	de la demanda durante el tiempo de	máxima admisible de que se agote la	35,01	150,01
		1	115	20	0,040		
3	Parametros	la colocación y la recepción de un pedido	promedio durante el	de la demanda durante el tiempo de	máxima admisible de que se agote la	121,99	1105,99
		8	123	21	0,020		
4	Parametros	la colocación y la recepción de un pedido	promedio durante el	de la demanda durante el tiempo de	máxima admisible de que se agote la	97,71	1636,71
		9	171	14	0,010		
5	Parametros	la colocación y la recepción de un pedido	promedio durante el	de la demanda durante el tiempo de	máxima admisible de que se agote la	87,43	1295,43
		8	150	22	0,080		
6	Parametros	la colocación y la recepción de un pedido	promedio durante el	de la demanda durante el tiempo de	máxima admisible de que se agote la	149,28	1024,28
		7	125	30	0,030		
7	Parametros	la colocación y la recepción de un pedido	promedio durante el	de la demanda durante el tiempo de	máxima admisible de que se agote la	60,33	1680,33
		9	180	15	0,090		
8	Parametros	la colocación y la recepción de un pedido	promedio durante el	de la demanda durante el tiempo de	máxima admisible de que se agote la	34,86	778,86
		4	186	13	0,090		
9	Parametros	la colocación y la recepción de un pedido	promedio durante el	de la demanda durante el tiempo de	máxima admisible de que se agote la	67,7	704,7

SIMULACIÓN DE DATOS

Modelo con preparación (política s-S)

	Cantidad a la mano, antes de	Costo de preparación	Costo de compra por unidad	Costo de almacenamiento por	Penalización por unidad faltante	Demanda máxima (D_max)	S	s1	s2	Cantidad a pedir
	5	24,72	2,04	2,86	18,3	94	7,68	17,27	-1,91	No pedir
	29	63,19	4,54	0,78	12,2	26	15,34	31,63	0,28	No pedir
	21	46,5	8,61	3,81	14,29	20	6,27	16,47	-3,85	No pedir
	6	30,32	3,95	3,08	6,46	23	6,05	18,58	-6,03	No pedir
	22	59,36	5,29	4,88	12,96	12	5,15	14,13	-3,77	No pedir
	30	78,71	4,27	1,76	15,9	73	48,07	74,07	22,84	18,07
	24	62,23	4,42	2,67	6,1	70	13,4	45,22	-18,13	No pedir
	3	26,39	4,3	7,3	13,13	10	4,32	9,41	-0,76	No pedir
	13	58,77	1,37	3,09	6,9	50	27,67	56,45	4,98	14,67
	15	26,97	8,26	3,17	13,13	63	18,82	33,33	4,41	3,82
	6	48,4	8,69	4,95	16,3	60	21,48	38,03	4,96	15,48

Figura 15. Simulación de datos modelos probabilizado y con política S-s

En estas figuras se puede apreciar de manera más clara lo que ha sido previamente explicado. Se han tomado en cuenta tanto modelos determinísticos como probabilísticos con el fin de que se pueda identificar si se ha aplicado la misma metodología en cada uno de los diferentes modelos. A través de estas ilustraciones, es posible observar que únicamente se han registrado los parámetros y resultados correspondientes, independientemente del tipo de modelo utilizado, omitiendo el detalle del proceso necesario para su cálculo. Esta presentación concisa permite centrar la atención en los datos clave obtenidos sin entrar en la descripción técnica de los pasos intermedios, facilitando así una visión general comparativa de los resultados entre los distintos modelos.

Para facilitar una interpretación más clara y un entendimiento profundo de este proceso, se ha incluido en el Anexo 8 un registro exhaustivo de las simulaciones realizadas para cada uno de los códigos y modelos presentados. Este anexo contiene de manera detallada todas las ejecuciones, resultados y análisis correspondientes, ofreciendo una visión integral del comportamiento de los modelos bajo diversas condiciones.

4.1.4.6. Creación del programa principal

Una vez que se completaron las bases de datos necesarias para el funcionamiento del programa principal, se inició el proceso de implementación de inteligencia artificial utilizando árboles de decisión. Inicialmente, se había tomado la decisión de emplear exclusivamente árboles de predicción para todos los modelos del sistema.

Sin embargo, a medida que avanzaba el desarrollo del programa, se evaluó la necesidad de diversificar las herramientas de modelado. Esto llevó a la decisión de utilizar tanto árboles de predicción como árboles de clasificación, seleccionando el tipo de árbol más adecuado según las características específicas de cada modelo. Esta flexibilidad permite un mejor ajuste a las diversas situaciones y requisitos que pueden surgir en el análisis de datos, optimizando así el rendimiento del sistema en su conjunto.

4.1.4.6.1. Árbol de decisión por predicción

Para la creación del programa basado en árboles de decisión para predicción, se emplearon diversas bibliotecas y códigos. A continuación, se detalla el proceso paso a paso que ilustra el funcionamiento del programa y las distintas operaciones realizadas. En esta sección, se utilizará el modelo *EOQ* como base para explicar su funcionamiento.

El código creado se utiliza para predecir la cantidad óptima de pedido en un modelo de inventario basado en la cantidad económica de pedido (*EOQ*).

```
[ ]  import numpy as np
     import pandas as pd
```

Figura 16. Bibliotecas usadas

Para comenzar, como se observa en la Figura 16, se importa *NumPy*, que facilita la manipulación de arreglos multidimensionales y la realización de operaciones matemáticas complejas. En este contexto, *NumPy* se utiliza para transformar los datos en arreglos numéricos, lo que permite una manipulación eficiente en las operaciones de aprendizaje automático. De igual forma se importa la biblioteca *Pandas*, que es esencial para la manipulación y análisis de datos estructurados.

```
[ ]  data = pd.read_excel("/content/drive/MyDrive/Modelos de inventarios.xlsx",sheet_name="EOQ")
     data
```

Figura 17. Lectura de datos

```
[ ]  data = data.dropna()
```

Figura 18. Limpieza de datos

En la Figura 17 y 18, el código utiliza *Pandas* para leer un archivo de *Excel* que en este caso es la base de datos que ya se había realizado con los programas que se desarrollaron anteriormente, dicha base de datos tiene todas las simulaciones por cada modelo, cada modelo en una hoja del *Excel*, entonces en este caso específicamente, se accede a la hoja llamada "*EOQ*" y se eliminan las filas con valores faltantes mediante la función *dropna* (), asegurando así que se trabajará únicamente con datos completos y precisos.

	D	K	h	L	cantidad_pedido	reorden
0	100	100.0	0.02	12	1000	200
1	300	20.0	0.21	1	239	60
2	120	30.0	0.04	5	424	175
3	145	23.0	0.05	6	365	139
4	80	15.0	0.03	10	282	234
...	...	...	...	...	...	...
95	128	149.5	0.06	8	798	225
96	314	118.2	6.31	1	108	97
97	910	19.6	3.70	5	98	33
98	680	79.2	0.25	2	656	47
99	150	144.3	0.09	4	693	600

100 rows × 6 columns

```
features = np.asarray(data[["D", "K", "h", "L"]])
targets = np.asarray(data["cantidad_pedido"])
```

Figura 19. Selección de Características y Objetivos

En la Figura 19 se visualiza que para este caso las características del modelo incluyen la demanda (D), el costo de pedido (K), el costo de mantenimiento (h) y el tiempo de entrega (L), se seleccionan y convierten en un arreglo de *NumPy*. Este arreglo se utilizará como las entradas del modelo de aprendizaje automático. Por otro lado, la columna que representa El lote óptimo de pedido se convierte en la variable objetivo que el modelo intentará predecir.

```
[ ]  from sklearn.tree import DecisionTreeRegressor
     from sklearn.model_selection import KFold
     k = 5
     kf = KFold(n_splits = k, shuffle = True)
     d_tree = DecisionTreeRegressor(max_depth= 50)
```

Figura 20. Implementación modelo de regresión

En la Figura 20, se implementa el modelo de regresión, primero se importa el árbol de decisión de la biblioteca *Scikit-learn*. Se crea un modelo de árbol de decisión, especificando un parámetro de profundidad máxima de 50. Este límite se establece para evitar el sobreajuste, un fenómeno que ocurre cuando un modelo es tan complejo que aprende detalles y ruido del conjunto de datos en lugar de generalizar adecuadamente a nuevos datos. En este contexto, el árbol de decisión toma decisiones basadas en las características de entrada para estimar la cantidad de pedido.

La validación cruzada *K-Fold* se utiliza como técnica para evaluar el modelo de manera más robusta. En este caso, el conjunto de datos se divide en cinco subconjuntos. Durante cada iteración, uno de estos subconjuntos se utiliza como conjunto de prueba, mientras que los otros cuatro se utilizan para entrenar el modelo. Este proceso se repite para garantizar que cada subconjunto de datos se evalúe como conjunto de prueba en algún momento. La opción de mezclar los datos aleatoriamente asegura que no haya sesgos en la distribución de los datos.

```
all_predictions = np.zeros_like(targets)
```

Figura 21. Almacenamiento de predicciones

Se crea un arreglo vacío del mismo tamaño que la variable objetivo para almacenar las predicciones generadas en cada iteración de validación cruzada, en la Figura 21.

```
# declarar un array para los valores predichos en cada iteración

all_predictions = np.zeros_like(targets)

for train_indices, test_indices in kf.split(features):
  train_features, test_features = features[train_indices], features[test_indices]
  train_targets, test_targets = targets[train_indices] , targets[test_indices]

  ## Entrenar el decision tree
  d_tree.fit(train_features,train_targets)

  ## Predecir usando el modelo
  predictions = d_tree.predict(test_features)

  ## Ponemos las predicciones en el array de all_predictions creado above
  all_predictions[test_indices] = predictions
```

Figura 22. *Bucle* que entrena el modelo

En la Figura 22, se crea un *bucle* que itera sobre los cinco subconjuntos de datos generados por *K-Fold*. En cada iteración, los datos se dividen en un conjunto de entrenamiento, que incluye las características de entrenamiento y los objetivos de entrenamiento, y un conjunto de prueba, compuesto por las características de prueba y los objetivos de prueba. A continuación, se entrena el modelo de árbol de decisión utilizando los datos de entrenamiento, y luego se realizan predicciones para el conjunto de prueba empleando las características de prueba. Las predicciones se almacenan en la variable *all_predictions* en las posiciones correspondientes a los índices del conjunto de prueba.

```
## Evaluamos el modelo usando la diferencia de medianas de todas las predicciones realizadas
eval_model = np.median(np.abs(all_predictions - targets))
eval_model
```

Figura 23. Evaluación del modelo

En la figura 23, el código evalúa el rendimiento del modelo, se calcula la mediana de las diferencias absolutas entre las predicciones generadas y los valores reales de la variable objetivo. Esta medida proporciona una indicación clara de qué tan cerca están las predicciones del modelo de los valores reales.

```python
from matplotlib import pyplot as plt
plt.figure(figsize = (3,3))
plt.scatter(targets, all_predictions)
plt.xlim(30,targets.max())
plt.ylim(30,predictions.max())
plt.xlabel("valores reales")
plt.ylabel("valores predichos")
plt.show()
```

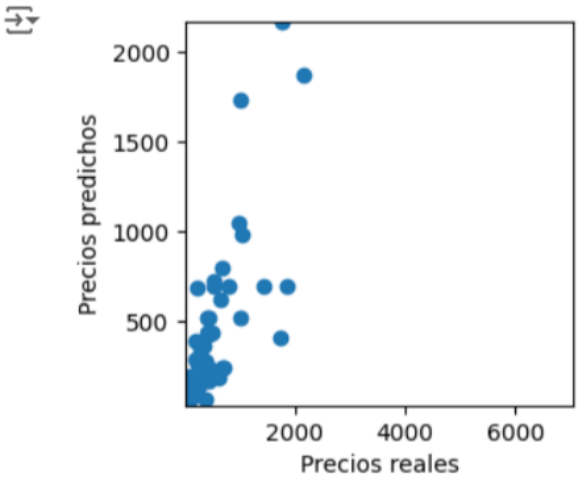


Figura 24. Gráfico dispersión

En la figura 24, el código utiliza la biblioteca *Matplotlib* para visualizar los resultados mediante un gráfico de dispersión que compara los valores reales de las predicciones (*targets*) con las predicciones generadas por el modelo (*all_predictions*). Esto permite evaluar visualmente el rendimiento del modelo y observar qué tan cercanas están las predicciones a los valores reales.

```
[ ]  cop = d_tree.predict([[900, 30, 0.01, 3]])

[ ]  cop

     array([534.])
```

Figura 25. Predecir valores

Finalmente, en la Figura 25, el código utiliza el modelo de árbol de decisión entrenado para hacer una predicción sobre un nuevo conjunto de características. Aquí, se le da al modelo una demanda de 900, un costo de pedido de 30, un costo de mantenimiento de 0.01, y un tiempo de entrega de 3. El modelo devuelve la cantidad óptima de pedido predicha para estas condiciones específicas.

Así concluye el programa para el cálculo del modelo determinístico *EOQ* (Cantidad Económica de Pedido). En este ejemplo, el código fue utilizado para determinar la cantidad económica de pedido; sin embargo, este mismo código puede emplearse para calcular el punto de reorden. Para ello, simplemente es necesario modificar la columna objetivo o "*targets*". Una vez que esta columna se ajuste para calcular el punto de reorden, el programa hará las predicciones correspondientes. Este código también es adaptable para diferentes modelos, tanto probabilísticos como determinísticos. La única modificación requerida es cambiar la hoja de *Excel* que el programa utiliza para obtener los datos. Al hacer esto, se actualizan tanto las entradas como las salidas que el programa procesa. Una vez definidas estas configuraciones, este código basado en el árbol de decisión de predicción se vuelve utilizable para la variedad de modelos tanto determinísticos como probabilísticos.

Determinísticos:

- Modelo clásico de cantidad económica de pedido
- Cantidad económica de pedido con discontinuidades de precio
- Cantidad económica de pedido de varios artículos con limitación de almacén
- Modelo dinámico sin costo de preparación
- Modelo dinámico con preparación

Probabilísticos:

- Modelo "probabilizado" de cantidad económica de pedido
- Modelo probabilista de cantidad económica de pedido
- Modelo de un solo periodo sin preparación

Una vez definido cómo funciona el código y con qué modelos es compatible, es importante abordar un aspecto clave del problema, que son los modelos que manejan múltiples artículos y periodos. Estos modelos calculan datos específicos para cada periodo o artículo individualmente, lo que significa que no es posible predecir todos los resultados de una sola vez. Para resolver este desafío, se implementó una solución sencilla basada en la comprensión de cómo se resuelve cada uno de los modelos de inventario.

Tras investigar a fondo la resolución de estos modelos, se identificó que, en aquellos con periodos y artículos múltiples, los valores dependen del resultado calculado en el periodo o artículo anterior. Por ejemplo, en el modelo dinámico sin costo de preparación, para calcular la cantidad a pedir en el periodo adicional 1, primero se

debe calcular la cantidad correspondiente al periodo normal 1, y así sucesivamente hasta completar todos los periodos. Entendiendo esta lógica, la solución adoptada fue ejecutar el código varias veces, ajustando el número de entradas conforme aumentaban los periodos o el número de artículos. De esta forma, se asegura que cada cálculo sea correcto y se realice en el orden adecuado, permitiendo solventar todos los periodos y obtener los resultados precisos para cada artículo o periodo.

	articulo.1	articulo.2	articulo.3	D1	D2	D3	K1	K2	K3	h1	h2	h3	a1	a2	a3	A	pedido_1	pedido_2	pedido_3	area_total
0	1	2	3	2	4	4	10	5	15	0.3	0.1	0.2	1	1	1	25	6.34	7.09	11.57	24.999
1	1	2	3	3	2	9	11	16	9	0.4	0.7	0.7	5	4	3	65	4.45	4.59	8.12	64.956
2	1	2	3	3	3	4	16	7	18	0.7	0.7	0.2	3	4	1	46	6.10	3.62	13.20	45.993
3	1	2	3	9	10	3	8	19	7	0.4	0.6	0.4	5	5	5	72	4.58	7.34	2.47	71.962
4	1	2	3	8	4	7	17	13	16	0.3	0.2	0.3	4	2	4	74	7.90	6.83	7.17	73.957
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
95	1	2	3	4	9	4	15	18	16	0.3	0.5	0.5	2	4	2	50	5.75	6.73	5.78	49.983
96	1	2	3	6	4	7	7	14	15	0.7	0.7	0.3	2	2	3	25	3.01	3.48	4.00	24.995
97	1	2	3	9	8	4	5	18	18	0.1	0.5	0.4	4	2	5	43	2.98	7.22	3.33	42.987
98	1	2	3	8	2	9	17	12	8	0.8	0.7	0.1	4	2	4	26	3.20	1.88	2.36	26.000
99	1	2	3	5	10	4	16	6	8	0.6	0.6	0.8	1	5	3	40	9.03	4.02	3.62	39.984

100 rows × 20 columns

```
[ ] data = data.dropna()

[ ] features = np.asarray(data[["D1", "D2", "D3", "K1", "K2", "K3", "h1", "h2", "h3", "a1", "a2", "a3", "A"]])
    targets = np.asarray(data["pedido_1"])
```

Figura 26. modelo con restricción de área pedido 1

```
features = np.asarray(data[["D1", "D2", "D3", "K1", "K2", "K3", "h1", "h2", "h3", "a1", "a2", "a3", "A","pedido_1"]])
targets = np.asarray(data["pedido_2"])
```

Figura 27. modelo con restricción de área pedido 2

```
[ ] features = np.asarray(data[["D1", "D2", "D3", "K1", "K2", "K3", "h1", "h2", "h3", "a1", "a2", "a3", "A","pedido_1","pedido_2"]])
    targets = np.asarray(data["pedido_3"])
```

Figura 28. modelo con restricción de área pedido 3

En la Figura 26 se presenta el código correspondiente al modelo con restricción de área, específicamente la parte que gestiona la selección de entradas y salidas. Como se puede observar, el programa recopila todos los datos necesarios para calcular los pedidos en la sección de "*features*" y establece como columna objetivo, o "*targets*", el valor de "pedido_1". En este caso, el programa predice la cantidad a solicitar para el primer artículo.

Tal como se explicó anteriormente, para calcular los valores correspondientes a los demás artículos, es necesario ejecutar el código nuevamente con una modificación en las entradas y salidas. En la Figura 27 se ilustra esta segunda ejecución del código, donde ahora "pedido_1" se incorpora como parte de las entradas, mientras que la nueva columna objetivo se ajusta a "pedido_2", con el fin de calcular el pedido para

el segundo artículo. Posteriormente, en la Figura 28, se presenta la siguiente ejecución del código, esta vez enfocada en el cálculo del pedido para el tercer artículo.

Este es el proceso que se debe seguir para realizar los cálculos correctamente mediante el programa. Si fuera necesario calcular pedidos para un mayor número de artículos o periodos que los ya establecidos en el programa, el proceso sería tan sencillo como agregar los datos correspondientes en la base de datos y subirla nuevamente al programa, permitiendo así que este opere con la cantidad de artículos o periodos requeridos.

Este tipo de manejo especial del programa es necesario para modelos que requieren cálculos específicos por artículo o por periodo. Entre los modelos que requieren este enfoque están los siguientes:

- Cantidad económica de pedido de varios artículos con limitación de almacén
- Modelo dinámico sin costo de preparación
- Modelo dinámico con preparación

Esta última sección concluye con la explicación completa y detallada del funcionamiento y los conocimientos necesarios para el correcto entendimiento y manejo del programa basado en árboles de decisión para predicción.

El proceso de modificación del código para calcular múltiples periodos o artículos ha sido explicado paso a paso, lo que permite una aplicación flexible y escalable del programa. Además, se ha aclarado cómo se puede fácilmente ajustar la base de datos para manejar mayores volúmenes de información o periodos adicionales, sin necesidad de reescribir por completo el código.

4.1.4.6.2. Árbol de decisión por clasificación

En esta sección, se explicará el funcionamiento del programa basado en árboles de clasificación. Sin embargo, es importante aclarar por qué se eligió este tipo de árbol en lugar de continuar con los árboles de regresión. Los árboles de regresión son útiles para predecir valores numéricos y para comprender la relación entre las características y la variable de respuesta numérica, lo que los convierte en la opción ideal en los modelos anteriores.

No obstante, en los modelos específicos que se están tratando aquí, a menudo es necesario que los resultados se expresen en términos cualitativos en lugar de

cuantitativos. Por esta razón, se optó por utilizar árboles de clasificación. Este tipo de árbol es especialmente eficaz para identificar categorías o clases, examinando también la relación entre las características y la respuesta. Así, el programa fue diseñado específicamente para abordar modelos que requieren este enfoque, permitiendo una mejor interpretación y comunicación de los resultados en situaciones donde las decisiones se basan en clasificaciones.

Este programa está diseñado específicamente para ser utilizado con modelos probabilísticos, enfocándose en dos:

- Modelo con preparación (política s-S)
- Modelos de varios periodos

La elección de aplicar este programa en estos modelos se fundamenta en la naturaleza de las respuestas que se generan. En ciertas situaciones, la respuesta a la decisión de pedido puede ser "No realizar el pedido". Esta respuesta, al ser cualitativa y no numérica, requiere un enfoque diferente para su análisis. Por lo tanto, el uso de árboles de decisión de clasificación resulta particularmente adecuado en este contexto.

Para este trabajo, se utilizará el modelo de varios periodos como base para explicar el funcionamiento de este programa. A continuación, se presentará un análisis detallado que describe paso a paso la estructura y el funcionamiento del código.

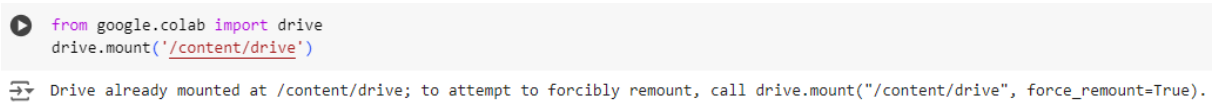


Figura 29. Montaje del Google *Drive*

El código comienza montando el Google *Drive* en el entorno de *Colab*. Como se visualiza en la Figura 29, esto se realiza en primer lugar importando la biblioteca *drive* y luego a través de la instrucción *drive.mount*('/content/*drive*'), permite acceder a los archivos del Google *Drive* dentro de *Colab*. Así, el archivo que contiene los datos del modelo de inventarios puede ser fácilmente cargado y manipulado más adelante en el programa.

```
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
%matplotlib inline
```

Figura 30. Importación de bibliotecas esenciales

Luego, en la Figura 30, el código realiza la importación de bibliotecas esenciales: *NumPy*, *pandas*, y *matplotlib*. *NumPy*, a través de su alias *np*, es utilizado para realizar operaciones matemáticas y manejar arreglos numéricos, mientras que *pandas*, con su alias *pd*, es importante para la manipulación de los datos tabulares en forma de *Dataframe*. *matplotlib.pyplot*, con su alias *plt*, se utiliza para la visualización de datos a través de gráficos, y el comando *%matplotlib inline* asegura que los gráficos se muestren directamente en el notebook después de su creación. Esto es importante para la depuración y el análisis visual de los resultados del modelo.

```
[ ] data = pd.read_excel("/content/drive/MyDrive/Modelos de inventarios.xlsx",sheet_name="M_varios_periodos")
    data
```

	x	p	α	r	c	h	D_max	nivel_optimo	cantidad_pedido
0	10	3	0.8	2	1	0.1	10	13.714286	Pedir
1	1	67	0.6	6	44	0.3	39	3.469154	Pedir
2	8	85	0.7	16	29	0.5	18	12.538206	Pedir
3	7	87	0.8	20	24	0.5	45	39.245902	Pedir
4	4	54	0.3	22	49	0.2	34	9.232759	Pedir
...	...	...	...	...	...	...	...	...	...
95	10	92	0.8	23	51	0.2	18	7.735537	No pedir
96	6	66	0.6	4	55	0.7	27	-6.166911	No pedir
97	4	81	0.4	5	44	0.2	41	12.855107	Pedir
98	9	69	0.8	24	35	0.7	27	17.830872	Pedir
99	11	58	0.4	9	54	0.5	33	-2.582160	No pedir

100 rows × 9 columns

Figura 31. Lectura de datos de un archivo *Excel*

El siguiente paso es la lectura del archivo *Excel* que contiene los datos del modelo, como se visualiza en la Figura 31. Esto utilizando *pd. read_excel()*. Aquí se especifica tanto la ruta al archivo en el Google *Drive* como la hoja específica que se va a leer, en este caso("M_varios_periodos"). Esto crea un *DataFrame* que contiene los datos que se usarán para entrenar y evaluar el modelo.

```
data.describe()
```

	x	p	α	r	c	h	D_max	nivel_optimo
count	100.000000	100.000000	100.000000	100.000000	100.000000	100.000000	100.000000	100.000000
mean	7.590000	63.480000	0.557000	18.310000	40.970000	0.421000	29.800000	10.766950
std	3.088248	20.996623	0.170712	17.195662	12.940778	0.227989	12.398273	13.181337
min	1.000000	3.000000	0.300000	2.000000	1.000000	0.100000	10.000000	-44.074074
25%	5.000000	47.750000	0.400000	8.750000	30.750000	0.200000	18.000000	3.317382
50%	8.000000	59.500000	0.600000	13.500000	42.000000	0.400000	30.000000	9.507723
75%	10.000000	83.000000	0.700000	22.000000	52.000000	0.600000	39.250000	18.367522
max	13.000000	100.000000	0.800000	91.000000	88.000000	0.800000	50.000000	49.628844

Figura 32. Análisis básico de los datos

Después de cargar los datos, en la Figura 32, se exploran con la función *data.describe(),* que genera estadísticas descriptivas como la media, desviación estándar, mínimos y máximos de las columnas numéricas. Esta exploración es fundamental para tener una idea inicial de los datos y su distribución antes de proceder con el modelado.

```
data["cantidad_pedido"].value_counts()
```

cantidad_pedido	count
Pedir	56
No pedir	44

dtype: int64

Figura 33. Análisis columna objetivo

También en la Figura 33, se realiza un análisis específico de la columna "cantidad_pedido", utilizando data["cantidad_pedido"].value_counts (), que cuenta la frecuencia de los valores "Pedir" y "No pedir". Este análisis ayuda a entender la distribución del objetivo de la predicción.

```
[ ] data["cantidad_pedido"].unique()

array(['Pedir', 'No pedir'], dtype=object)

[ ] Nopedir = {value: key for key,value in enumerate(data["cantidad_pedido"].unique())}

[ ] data["cantidad_pedido"] = data["cantidad_pedido"].map(Nopedir)

gender_label = {value: key for key,value in enumerate(data["cantidad_pedido"].unique())}
data["cantidad_pedido"] = data["cantidad_pedido"].map(gender_label)
data
```

	x	p	α	r	c	h	D_max	nivel_optimo	cantidad_pedido
0	10	3	0.8	2	1	0.1	10	13.714286	0
1	1	67	0.6	6	44	0.3	39	3.469154	0
2	8	85	0.7	16	29	0.5	18	12.538206	0
3	7	87	0.8	20	24	0.5	45	39.245902	0
4	4	54	0.3	22	49	0.2	34	9.232759	0
...	...	...	...	...	...	...	...	...	...
95	10	92	0.8	23	51	0.2	18	7.735537	1
96	6	66	0.6	4	55	0.7	27	-6.166911	1
97	4	81	0.4	5	44	0.2	41	12.855107	0
98	9	69	0.8	24	35	0.7	27	17.830872	0
99	11	58	0.4	9	54	0.5	33	-2.582160	1

100 rows × 9 columns

Figura 34. *Mapeo* de valores de texto a numéricos

En la Figura 34, se encuentra todo el proceso para convertir los valores de texto a números, en primer lugar, se obtiene los valores únicos de la columna cantidad_pedido, devolviendo las categorías que se encuentran en dicha columna, que en este caso son ['Pedir', 'No pedir']. La función *unique* () ayuda a identificar las distintas categorías o clases.

En la siguiente línea de código, se creó un diccionario llamado "Nopedir", donde a cada valor de la columna cantidad_pedido se le asigna un número. La función *enumerate()* asigna un índice numérico a cada valor único de cantidad_pedido. El resultado será un diccionario como { 'Pedir': 0, 'No pedir': 1 }, donde "Pedir" se mapea al valor 0 y "No pedir" al valor 1.

En la línea data["cantidad_pedido"] = data["cantidad_pedido"].map(Nopedir), se reemplazan las etiquetas originales de la columna cantidad_pedido por los valores numéricos definidos en el diccionario Nopedir. Esto se realizó por lo que es necesario

convertirlas en valores numéricos. Una vez realizado esto en la última línea de código de esta figura, se genera la tabla, pero ahora con los valores cambiados a 0 y 1.

```
data["cantidad_pedido"].value_counts()
```

cantidad_pedido	count
0	56
1	44

dtype: int64

Figura 35. Análisis columna objetivo-nueva

Con la nueva tabla, en la Figura 35, se vuelve a utilizar la función *value_counts()*, pero esta vez para contar la frecuencia de los valores numéricos 0 y 1, que ahora representan las decisiones de "No pedir" y "Pedir" respectivamente. Esto permite obtener una visión clara de cuántas veces se presenta cada una de estas categorías en los datos transformados.

```
import seaborn as sns
plt.style.use("default")
plt.figure(figsize=(10,10))
sns.heatmap(data.corr(),annot = True)
plt.title("Correlation Matrix")
plt.show()
```

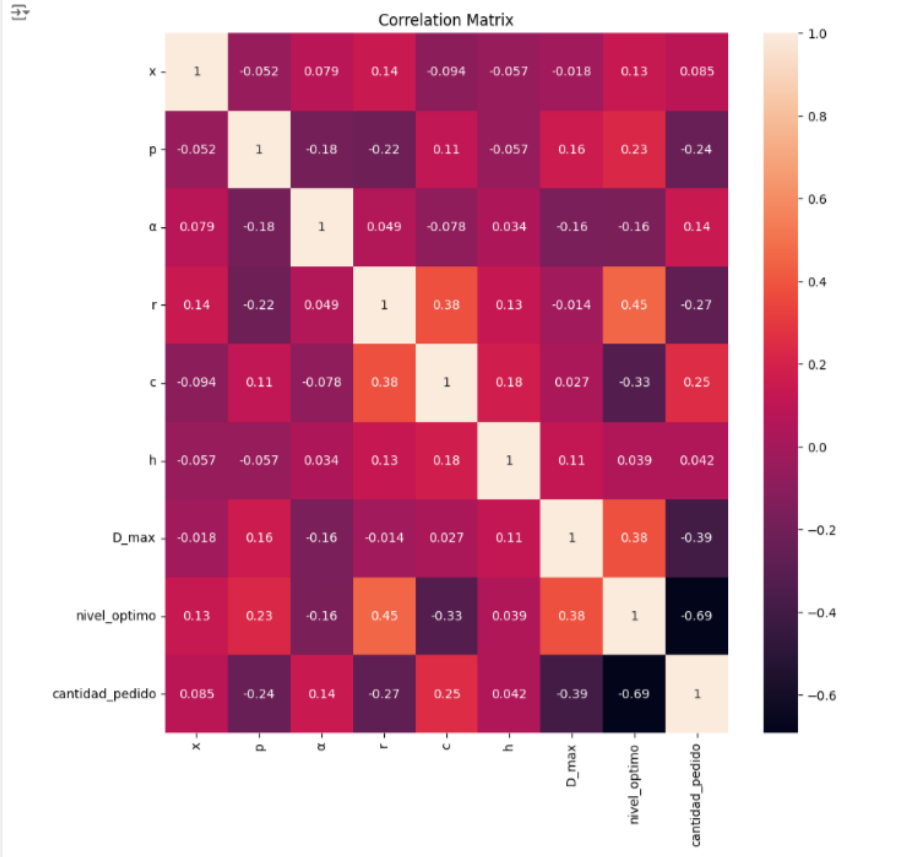


Figura 36. Matriz de correlación

A continuación, en la Figura 36, se genera una visualización de la matriz de correlación entre las variables del *DataFrame* utilizando la biblioteca *seaborn* para crear un mapa de calor. Esta matriz de correlación permite identificar qué variables están más relacionadas entre sí, lo que es útil para mejorar el rendimiento del modelo eliminando variables redundantes o seleccionando las más relevantes.

```
features = np.asarray([data["x"],
                       data["p"],
                       data["a"],
                       data["r"],
                       data["c"],
                       data["h"],
                       data["D_max"],
                        data["nivel_optimo"]])

features = features.T
features
```

Figura 37. Preparación de las características

```
targets = np.asarray(data["cantidad_pedido"])

features.shape,targets.shape
```

```
((100, 8), (100,))
```

Figura 38. Preparación de los objetivos

En la Figura 37 y la Figura 38, se determinan las entradas y objetivos del programa, en primer lugar, la función *np. asarray()* se utiliza para convertir los datos de *pandas* en arreglos de *NumPy*, lo que facilita el manejo eficiente de grandes volúmenes de datos. La transposición de la matriz de características se realiza con *features*. T, de manera que cada fila represente un conjunto de datos correspondiente. El arreglo *features* contiene las columnas relevantes para el modelo de predicción, que incluyen variables como x, p, a, r, c, h, D_max y nivel_optimo. Por otro lado, el arreglo targets contiene la columna cantidad_pedido, que será el objetivo de la predicción. Así, tanto las características (variables independientes) como los objetivos (variables dependientes) se preparan mediante arreglos *NumPy*, asegurando que las características se estructuren adecuadamente y que la columna cantidad_pedido se utilice como la variable a predecir.

```
[ ] from sklearn.model_selection import train_test_split
    train_features,test_features,train_targets,test_targets = train_test_split(features,targets,random_state = 60,test_size=0.2)
    train_features.shape,test_features.shape
```

Figura 39. División de datos en conjuntos de entrenamiento y prueba

En esta parte del código que se ve en la Figura 39, se utiliza la función *train_test_split()* para dividir los datos en dos conjuntos: el 80% de los datos se destina al entrenamiento del modelo y el 20% restante para su prueba. Los argumentos clave incluyen *random_state*=60, que garantiza la reproducibilidad del proceso de división, y *test_size*=0.2, que especifica que el 20% de los datos se reservará para la prueba. Este procedimiento asegura que el modelo se entrene con la mayoría de los datos disponibles, pero que también sea validado con un conjunto independiente, lo que permite evaluar su rendimiento de manera más objetiva.

```
[ ]  from sklearn.preprocessing import StandardScaler
     sc = StandardScaler()
     train_features = sc.fit_transform(train_features)
     test_features = sc.fit_transform(test_features)
```

Figura 40. Estandarización de los datos

Antes de entrenar el modelo, en la Figura 40, los datos son estandarizados utilizando *StandardScaler()*. La estandarización transforma los datos para que tengan una media de 0 y una desviación estándar de 1. Esto es especialmente útil para algoritmos que son sensibles a las escalas de los datos, como los árboles de decisión en este caso.

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix

d_tree = DecisionTreeClassifier(max_depth=5)

d_tree
```

```
DecisionTreeClassifier
DecisionTreeClassifier(max_depth=5)
```

Figura 41. Creación modelo árbol de decisión

En la Figura 41, el modelo de árbol de decisión se crea utilizando *DecisionTreeClassifier()* de la biblioteca *sklearn*. Aquí, se define una profundidad máxima de 5 para evitar el sobreajuste del modelo (es decir, que el árbol sea demasiado complejo y se ajuste excesivamente a los datos de entrenamiento).

```
d_tree.fit(train_features,train_targets)
```

```
DecisionTreeClassifier
DecisionTreeClassifier(max_depth=5)
```

Figura 42. Entrenamiento del modelo de árbol de decisión

Luego, en la Figura 42, el modelo se entrena con el conjunto de datos de entrenamiento utilizando *d_tree.fit()*.

```
predictions = d_tree.predict(test_features)

[ ] print(classification_report(test_targets,predictions))

              precision    recall  f1-score   support

           0       1.00      0.83      0.91        12
           1       0.80      1.00      0.89         8

    accuracy                           0.90        20
   macro avg       0.90      0.92      0.90        20
weighted avg       0.92      0.90      0.90        20
```

Figura 43. Predicción y evaluación del modelo

Como se evidencia en la Figura 43, la función *predict()* utiliza el modelo previamente entrenado para predecir los valores del conjunto de prueba, mientras que *classification_report()* proporciona un resumen detallado de las métricas de rendimiento del modelo, como la precisión, el *recall* y el puntaje F1, lo que permite evaluar su efectividad al comparar las predicciones con los resultados reales del conjunto de prueba.

```
accuracy_score(test_targets,predictions)

0.9

[ ] np.median(np.abs(predictions-test_targets))

0.0
```

Figura 44. Precisión del modelo

En la Figura 44, se establecen los códigos para evaluar la precisión del modelo. La función *accuracy_score(test_targets, predictions)* mide la precisión del modelo al calcular la proporción de predicciones correctas comparadas con el total de predicciones realizadas. Por otro lado, *np.median(np.abs(predictions - test_targets))* calcula la mediana del error absoluto entre las predicciones del modelo y los valores reales, lo que indica el error típico en las predicciones sin ser afectado por valores extremos. Ambos sirven para evaluar el rendimiento del modelo, uno midiendo la exactitud y el otro el error típico de las predicciones.

```
plt.figure(figsize=(5,5))
sns.heatmap(confusion_matrix(test_targets,predictions),annot=True)
plt.title("Predictions vs. targets")
plt.show()
```

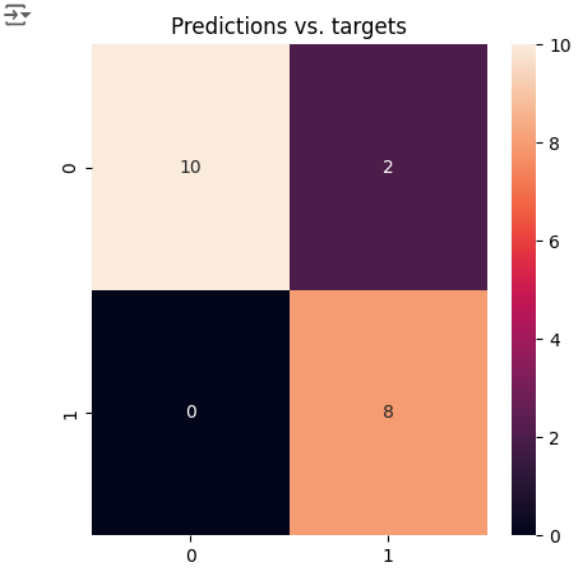


Figura 45. Matriz de confusión con valores predichos

Como se observa en la Figura 45, en esta parte del programa se crea una matriz de confusión que muestra la comparación de los valores reales con los predichos.

```
from sklearn import tree
fig = plt.figure()
_ = tree.plot_tree(d_tree,filled = True)
```

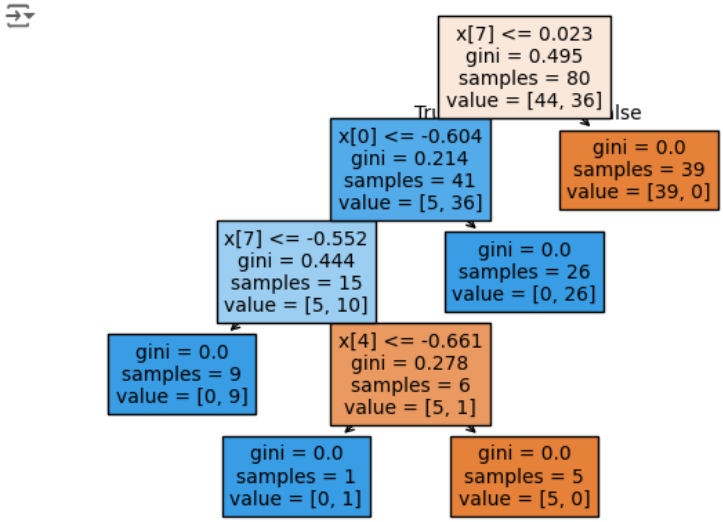


Figura 46. Visualización del árbol de decisión

En este fragmento de código de la Figura 46, se utiliza la biblioteca *sklearn* para visualizar un árbol de decisión entrenado. En primer lugar, se importa el módulo *tree* de *sklearn*, que contiene herramientas para crear y manipular árboles de decisión. Luego, se crea una nueva figura con *plt.figure()* para definir el espacio donde se

generará la visualización. Finalmente, con la función *tree.plot_tree(),* se dibuja el árbol de decisión almacenado en la variable *d_tree,* utilizando la opción *filled=True*, que rellena los nodos del árbol con colores que representan la pureza de las clases en cada nodo, lo que facilita la interpretación visual del modelo.

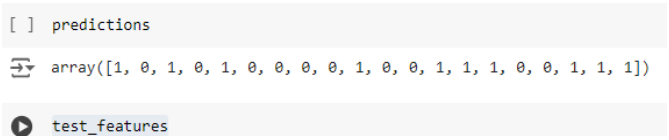


Figura 47. Resultado predicciones

En la Figura 47, se visualiza las líneas de código para el resultado de las predicciones, *predictions* es el resultado de las predicciones realizadas por el modelo de árbol de decisión (*d_tree*) en los datos de prueba. Por otro lado, *test_features* es el conjunto de características que se ha dividido del conjunto total de datos. Este conjunto es utilizado para probar el modelo, es decir, se usa para predecir los valores objetivo (en este caso, "Pedir" o "No pedir") sin que el modelo haya visto estos datos durante el entrenamiento.

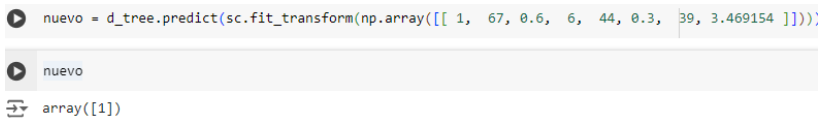


Figura 48. Predicción del modelo

Finalmente, en la última parte del código en la Figura 48, se está utiliza el modelo entrenado *(d_tree)* para hacer una predicción con un nuevo conjunto de características que se ingresan manualmente en forma de un arreglo. Primero, se crea un arreglo de *NumPy* con los valores *np.array*([[1 , 67 , 0,6 , 6 , 44 , 0.3 , 39 , 3,469154]]), donde cada uno representa una característica específica, como el inventario inicial "x" (1), costo de penalización "p" (67), factor de descuento "α" (0,6), precio de venta por unidad "r" (6), precio de comprar por unidad "c" (44), costo de almacenamiento "h" (0,3), intervalo máximo de demanda "*D_max*" (39) y nivel óptimo (3,469154). Luego, *sc.fit_transform()* estandariza estas nuevas características de entrada para asegurarse de que estén en la misma escala que los datos con los que se entrenó el modelo. Posteriormente, *d_tree.predict()* utiliza el modelo entrenado para predecir si, con base en estas características, se debe "Pedir" (1) o

"No pedir" (0). El resultado se almacena en la variable nuevo, y en este caso, la predicción es "Pedir" (1).

Así concluye el programa. Sin embargo, como se ha mencionado, este código está diseñado para determinar si se debe "pedir" o "no pedir". El modelo clasifica estos dos resultados; sin embargo, si la respuesta es "pedir", el programa no calcula la cantidad que se debe solicitar. Para abordar este inconveniente, se desarrolló el programa utilizando el modelo de árbol de decisión de predicción, también aplicable a los modelos probabilísticos S-s y a los de varios periodos. En estos programas, ya es posible calcular la cantidad a pedir, ya que esta se considera una variable numérica.

Una vez finalizada la explicación de los distintos programas basados en árboles de decisión, tanto para predicción como para clasificación, es fundamental destacar que el código de cada uno de los modelos se encuentra detallado en el Anexo 8. Este anexo documenta en su totalidad el programa, aquí se podrá obtener una comprensión más detallada del proceso de implementación y la lógica detrás de cada modelo.

4.1.5. Realizar una simulación de los procesos de gestión de inventarios con el programa ya implementado

Una vez desarrollados los programas para el cálculo de los modelos determinísticos y probabilísticos, se procedió a realizar simulaciones con el fin de evaluar su efectividad. Aunque cada programa, ya sea de decisión o de clasificación, proporciona métricas que indican la precisión del modelo en sus predicciones, resulta esencial realizar una comparativa de los resultados obtenidos con estos programas y contrastarlos con aquellos generados por el programa que utiliza fórmulas matemáticas exactas para el cálculo preciso de los modelos.

Para llevar a cabo esta evaluación, se seleccionaron cuatro modelos distintos: dos determinísticos y dos probabilísticos. A continuación, se procedió a calcular los resultados utilizando tanto el programa basado en árboles de decisión como el programa que ofrece el resultado exacto. Esto permitió una comparación detallada entre ambos enfoques, con el objetivo de medir la precisión y fiabilidad de los modelos predictivos frente a los cálculos exactos. Este análisis resulta clave para validar la utilidad y efectividad de las aproximaciones algorítmicas frente a los métodos tradicionales de cálculo exacto.

4.1.5.1. Modelos de inventarios determinísticos

En la Tabla 10, se llevó a cabo un análisis del modelo clásico de cantidad económica de pedido utilizando tanto el programa basado en árboles de decisión predictivos como el primer programa.

Tabla 10. Resultados comparativos *EOQ*: Árbol de Decisiones vs Programa

Modelo clásico de cantidad económica de pedido							
				Árbol de decisiones		Programa	
D	K	h	L	Cantidad pedido	Punto de reorden	Cantidad pedido	Punto de reorden
784	68,13	0,27	1	315,67	29,01	629,01	154,99
659	49,38	0,07	2	263,63	71,77	964,24	353,76
1753	90,26	2,50	10	258,79	201,88	355,78	96,70
1316	96,73	2,34	10	283,83	15,56	329,85	295,88
935	44,51	2,61	1	188,90	40,39	178,58	42,11
863	81,26	2,94	11	210,05	131,27	218,42	101,10
1831	57,60	2,40	5	210,38	138,51	296,46	711,50
1483	61,12	1,39	8	289,96	194,21	361,14	307,67
1133	93,30	2,34	4	286,30	237,97	300,58	23,36
1847	16,39	0,36	3	309,04	332,187	410,10	209,74

Se realizo esto con el objetivo de comparar los resultados generados por ambos enfoques. Para la comparación, se realizaron 10 simulaciones en cada programa, utilizando los mismos datos como base.

En la Tabla 11, se presentan las diferencias cuadráticas entre las predicciones de Cantidad de pedido y Punto de reorden generadas por dos métodos diferentes, permitiendo evaluar su precisión.

Tabla 11. MSE modelo clásico de cantidad económica de pedido

Diferencia cuadrática (cantidad de pedido)	Diferencia cuadrática (Punto de reorden)
98181,9556	15870,9604
490854,3721	79518,3601
9407,0601	11062,8324
2117,8404	78579,3024

	Diferencia cuadrática (cantidad de pedido)	Diferencia cuadrática (Punto de reorden)
	106,5024	2,9584
	70,0569	910,2289
	7409,7664	328317,5401
	5066,5924	12873,1716
	203,9184	46057,4521
	10213,1236	14,993
MSE	62363,11883	58818,60742

A pesar de que ya se observó de manera preliminar la diferencia entre los valores predichos por el árbol de decisión y los del primer programa desarrollado, se decidió realizar un análisis del error cuadrático medio (MSE). Esto permitió un entendimiento más estadístico sobre la dispersión de estos datos.

En la columna de Diferencia cuadrática (cantidad de pedido), se observan variaciones significativas, con el valor más bajo de 106,50 y el más alto de 490854,37. Estos valores indican que, aunque algunas predicciones son bastante cercanas, en otros casos existen discrepancias marcadas entre los resultados de los dos métodos. Esto sugiere que las predicciones de la cantidad de pedido son, en general, más consistentes, pero también presentan casos donde la diferencia es considerablemente alta.

Por otro lado, en la columna de Diferencia cuadrática (Punto de reorden), las diferencias también muestran variabilidad, oscilando entre un mínimo de 2,96 y un máximo de 328317,54. Al igual que en la cantidad de pedido, estas diferencias reflejan la disparidad en las predicciones, aunque los valores más altos en este caso no son tan extremos como los observados en la columna anterior.

Los Errores Cuadráticos Medios (MSE) calculados son 62363,12 para la Cantidad de pedido, este valor sugiere que, en promedio, la diferencia al cuadrado entre las cantidades a pedir predichas por el Árbol de decisiones y las cantidades a pedir del Programa es relativamente grande. Esto puede indicar que hay discrepancias significativas entre las cantidades que predice el modelo y las que realmente se utilizan.

Los errores cuadráticos medios para el punto de reorden son 58818,61 para el Punto de reorden, este valor indica la magnitud de los errores al cuadrado entre los puntos de reorden predichos por el modelo de Árbol de decisiones y los puntos de reorden reales del Programa. Un MSE de 58,818.61 sugiere que, en promedio, las predicciones del modelo se desvían significativamente de los valores reales.

En la Tabla 12, se llevó a cabo un análisis del modelo de cantidad económica de pedido con discontinuidad de precios utilizando tanto el programa basado en árboles de decisión predictivos como el primer programa

Tabla 12. Resultados Comparativos *EOQ* con descuento: Árbol de Decisiones vs Programa

Cantidad económica de pedido con discontinuidades de precio										
							Árbol de decisiones		Programa	
D	K	c1	c2	h	Q	L	Cantidad pedido	Punto de reorden	Cantidad pedido	Punto de reorden
137	528	8,44	6,63	0,11	953	1	1063,00	176,00	1146,00	137,00
335	410	13,02	9,06	0,15	332	4	1434,00	272,00	1353,00	1340,00
172	301	18,71	8,80	0,08	715	2	1063,00	92,00	1137,00	344,00
867	198	11,86	8,81	0,39	666	2	837,00	305,00	938,00	795,00
815	296	11,46	4,91	0,21	72	4	1052,00	1414,00	1515,00	228,00
745	274	19,09	3,41	0,03	827	3	3356,00	1161,00	3688,00	2235,00
947	377	16,68	7,69	0,32	757	1	1084,00	332,00	1493,00	947,00
244	488	15,74	7,15	0,33	176	1	507,00	171,00	849,00	244,00
229	272	17,04	7,70	0,16	329	3	1510,00	220,00	882,38	687,00
683	510	6,39	5,88	0,23	431	3	1610,00	595,00	1740,00	308,00

Esto se hizo con el objetivo de comparar los resultados generados por ambos enfoques. Para la comparación, se realizaron 10 simulaciones en cada programa, utilizando los mismos datos como base.

En la Tabla 13, se muestran las diferencias cuadráticas entre las predicciones de Cantidad de pedido y Punto de reorden generadas por dos métodos diferentes, permitiendo evaluar su precisión.

Tabla 13. MSE cantidad económica de pedido con discontinuidades de precio

	Diferencia cuadrática (cantidad de pedido)	Diferencia cuadrática (Punto de reorden)
	6889	1521
	6561	1140624
	5476	63504
	10201	240100
	214369	1406596
	110224	1153476
	167281	378225
	116964	5329
	393906,8644	218089
	16900	82,369
MSE	104877,1864	468983,3

En la columna de Diferencia cuadrática (cantidad de pedido), se observan variaciones que van desde un mínimo de 6,889 hasta un máximo de 393,906.86. Estos valores indican que, aunque algunas predicciones son relativamente cercanas, existen discrepancias notables en ciertos casos. Este rango sugiere que las predicciones de la cantidad de pedido pueden ser más consistentes en general, pero también muestran ejemplos de diferencias marcadas entre los resultados de los dos métodos.

Por otro lado, en la columna de Diferencia cuadrática (Punto de reorden), las diferencias reflejan una variabilidad aún más pronunciada, con un mínimo de 5,329 y un máximo de 1,400,596. Estas diferencias indican que las predicciones del Punto de reorden son más dispersas, con algunos casos en los que la discrepancia entre los dos métodos es extremadamente alta. Esto podría señalar que las predicciones del Punto de reorden son menos confiables y más susceptibles a variaciones significativas.

Los Errores Cuadráticos Medios (MSE) calculados son 104,877.19 para la Cantidad de pedido. Este valor sugiere que, en promedio, la diferencia al cuadrado entre las cantidades a pedir predichas por el Árbol de decisiones y las cantidades a pedir del Programa es considerablemente grande. Esto puede indicar que existen discrepancias notables entre las cantidades que predice el modelo y las que realmente se utilizan.

En cuanto a los errores cuadráticos medios para el punto de reorden, el valor es de 468,983.30. Este MSE elevado indica la magnitud de los errores al cuadrado entre los puntos de reorden predichos por el modelo de Árbol de decisiones y los puntos de reorden reales del Programa. Un MSE de 468,983.30 sugiere que, en promedio, las predicciones del modelo se desvían significativamente de los valores reales, lo que refleja la falta de precisión en las estimaciones del Punto de reorden en comparación con la Cantidad de pedido.

4.1.5.2. Modelos de inventarios probabilísticos

En la Tabla 14, se llevó a cabo un análisis del modelo probabilizado de cantidad económica de pedido utilizando tanto el sistema basado en árboles de decisión predictivos como el primer programa.

Tabla 14. Resultados Comparativos modelo probabilizado: Árbol de Decisiones vs Programa

Modelo "Probabilizado" de cantidad económica de pedido							
				Árbol de decisiones		**Programa**	
L	**D**	**σ**	**α**	**Punto de reorden**	***Stock* de seguridad**	**Punto de reorden**	***Stock* de seguridad**
2	159	35	0,039	537,58	137,77	295,59	72,99
6	113	29	0,023	897,22	133,18	819,74	141,74
7	168	36	0,049	1315,03	135,27	1333,60	157,60
8	193	31	0,011	1717,23	160,03	1744,82	200,82
9	155	23	0,034	1467,6	141,71	1520,93	125,93
8	186	39	0,015	1680,33	174,24	1727,38	239,38
2	121	27	0,044	313,9	53,4	307,14	65,14
1	174	33	0,018	235,53	59,56	243,20	69,20
6	142	25	0,027	918,27	133,18	969,99	117,99
1	108	38	0,041	150,01	84,9	174,09	66,09

Esto se hizo con el objetivo de comparar los resultados generados por ambos enfoques. Para la comparación, se realizaron 10 simulaciones en cada programa, utilizando los mismos datos como base.

En la Tabla 15, se muestran las diferencias cuadráticas entre las predicciones del Punto de reorden y el *Stock* de seguridad generadas por dos métodos diferentes, lo que permite evaluar su precisión y consistencia.

Tabla 15. MSE modelo "Probabilizado" de cantidad económica de pedido

	Diferencia cuadrática (punto de reorden)	**Diferencia cuadrática (*stock* de seguridad)**
	58559,1601	4196,4484
	6003,1504	73,2736
	344,8449	498,6289
	761,2081	1663,8241
	2844,0889	249,0084
	2213,7025	4243,2196
	45,6976	137,8276
	58,8289	92,9296
	2674,9584	230,7361
	579,8464	354
MSE	7408,54862	1173,97124

En la columna de Diferencia cuadrática (punto de reorden), se observa un rango de variabilidad que va desde un mínimo de 344,84 hasta un máximo de 58,559.16. Estos valores indican que, aunque algunas predicciones están relativamente cerca del valor real, existen discrepancias notables en ciertos casos. Esto sugiere que, en general, las estimaciones del Punto de reorden pueden ser inconsistentes, con algunos casos en los que la diferencia es considerablemente alta.

Por otro lado, en la columna de Diferencia cuadrática (*stock* de seguridad), las diferencias son notablemente más bajas, oscilando entre un mínimo de 73,27 y un máximo de 4,196.45. Esta menor variabilidad sugiere que las predicciones del *Stock* de seguridad son más consistentes y tienden a estar más cerca de los valores reales, reflejando una mayor fiabilidad en las estimaciones generadas por este método.

Los Errores Cuadráticos Medios (MSE) calculados son 7,408.55 para el Punto de reorden. Este valor indica que, en promedio, la diferencia al cuadrado entre los puntos de reorden predichos y los reales es relativamente alta, lo que puede señalar

la existencia de discrepancias significativas entre las predicciones del modelo y los valores utilizados.

En cuanto a los errores cuadráticos medios para el *Stock* de seguridad, el valor es de 1,173.97. Este MSE más bajo sugiere que, en promedio, las predicciones del *Stock* de seguridad se desvían menos de los valores reales en comparación con las predicciones del Punto de reorden. Esto refleja una mayor precisión en las estimaciones del *Stock* de seguridad, lo que implica que este método podría ser más efectivo para generar predicciones confiables en esta área.

En la Tabla 16, se llevó a cabo un análisis del modelo probabilístico de cantidad económica de pedido utilizando tanto el sistema basado en árboles de decisión predictivos como el primer programa.

Tabla 16. Resultados Comparativos modelo probabilístico: Árbol de Decisiones vs Programa

Modelo probabilístico de cantidad económica de pedido									
						Árbol de decisiones		Programa	
f(x)	D	K	h	p	E{x}	Cantidad pedido	Punto de reorden	Cantidad pedido	Punto de reorden
54	1127	94	2	10	54	394,632	32,073	327,05	50,86
43	1010	107	1	9	43	319,438	94,95	466,01	40,79
59	873	112	3	11	59	252,761	33,582	257,69	54,25
40	1189	87	27	20	40	478,586	57,447	89,59	35,93
146	958	101	1	12	146	249,67	7,85	442,72	140,37
55	1076	119	3	9	55	214,54	38,46	294,69	49,97
48	892	96	2	11	48	335,856	67,174	294,071	45,12
57	1145	103	1	10	57	301,67	58,599	486,87	54,57
52	1034	88	2	8	52	319,48	32,54	303,56	48,18
45	967	263	3	12	45	317,82	46,82	414,17	40,18

Esto se hizo con el objetivo de comparar los resultados generados por ambos enfoques. Para la comparación, se realizaron 10 simulaciones en cada programa, utilizando los mismos datos como base.

En la tabla 17, se muestran las diferencias cuadráticas entre las predicciones de la cantidad a pedir y el punto de reorden generadas por dos métodos distintos.

Tabla 17. MSE modelo probabilístico de cantidad económica de pedido

	Diferencia cuadrática (cantidad a pedir)	Diferencia cuadrática (punto de reorden)
	4567,326724	352,951369
	21483,35118	2933,3056
	24,295041	427,166224
	151317,888	462,981289
	37268,3025	17561,5504
	6424,0225	132,4801
	1745,986225	486,378916
	34299,04	16,232841
	253,4464	244,6096
	9283,3225	44
MSE	26666,69811	2266,174594

En la columna de Diferencia cuadrática (cantidad a pedir), se observan variaciones notables, con valores que van desde un mínimo de 24,29 hasta un máximo de 151,317.89. Estas diferencias reflejan una gran dispersión en las predicciones, lo que indica que, en algunos casos, las estimaciones de la cantidad a pedir presentan una considerable discrepancia con los valores reales. Este rango sugiere que las predicciones de la cantidad a pedir son menos consistentes y pueden generar errores significativos en determinados casos.

En la columna de Diferencia cuadrática (punto de reorden), las diferencias son menores y más consistentes, con un mínimo de 16,23 y un máximo de 17,561.55. Esto implica que las predicciones del punto de reorden tienden a ser más precisas, con menores diferencias en la mayoría de los casos, aunque existen algunos valores más altos que indican algunas desviaciones en ciertos puntos.

El Error Cuadrático Medio (MSE) para la cantidad a pedir es de 26,666.70, lo que indica que, en promedio, las diferencias cuadráticas entre la cantidad a pedir predichos y los reales son elevadas. Esto sugiere que las predicciones del modelo no son particularmente precisas y que hay una desviación significativa entre los valores estimados y los reales.

Por otro lado, el MSE para el punto de reorden es de 2,266.17, mucho menor en comparación con el MSE de cantidad a pedir. Esto implica que las predicciones del punto de reorden son, en promedio, mucho más precisas y están más cercanas a los valores reales, reflejando una mayor fiabilidad en las estimaciones generadas para este parámetro.

4.1.5.3. Propuesta de mejora del programa

Como se pudo observar en el punto anterior, a través de las tablas estadísticas realizadas, el programa no logra ofrecer predicciones exactas ni se aproxima significativamente a los resultados que deberían obtenerse con precisión. Los resultados obtenidos son considerablemente diferentes a los cálculos exactos, lo cual no sorprende dado que este programa se basa en predicción y clasificación. Exigirle una exactitud del 100% sería irracional, ya que no emplea fórmulas matemáticas en su núcleo, sino que depende de datos previamente calculados para su funcionamiento.

Es importante tener en cuenta que este tipo de programas están diseñados para ofrecer estimaciones, no resultados exactos. Por lo tanto, la falta de precisión absoluta es inherente a la naturaleza de los algoritmos de predicción. Sin embargo, se planteó la posibilidad de mejorar la capacidad predictiva del programa mediante la implementación de redes neuronales en lugar de árboles de decisión. Con esta premisa, se procedió a desarrollar una nueva versión del programa, esta vez utilizando redes neuronales como base, con el objetivo de aumentar la precisión de las predicciones y acercar los resultados a los cálculos exactos.

Este enfoque basado en redes neuronales ofrece la ventaja de poder aprender patrones complejos en los datos, lo que podría mejorar considerablemente la calidad de las predicciones, especialmente en escenarios donde los modelos basados en árboles de decisión presentan limitaciones. A través de esta mejora, se busca optimizar el rendimiento del programa, haciéndolo más robusto y preciso en la predicción de resultados frente a los modelos tradicionales.

4.1.5.3.1. Programa con redes neuronales

En esta sección se desarrolló el programa con redes neuronales, tanto para predicción como para clasificación. Se muestra el desarrollo del código paso a paso, junto con una explicación de cómo funciona cada parte de este.

4.1.5.3.1.1. Redes neuronales por predicción

Para proceder con el desarrollo del programa, se decidió utilizar como base el modelo clásico de la Cantidad Económica de Pedido (*EOQ*). Este modelo fue seleccionado por su amplia aplicabilidad en la gestión de inventarios y su relevancia

en la optimización de costos relacionados con pedidos y almacenamiento. Al tomar el *EOQ* como ejemplo, no solo se facilita el desarrollo del programa, sino que también se garantiza que el modelo elegido es uno de los más representativos y frecuentemente utilizados en la práctica empresarial.

```
[ ]  import numpy as np
     import pandas as pd
     from sklearn.model_selection import train_test_split
     from sklearn.preprocessing import StandardScaler
```

Figura 49. Importación de bibliotecas

Para empezar con la explicación del código, es importante destacar que este se realizará para el cálculo de la cantidad a pedir.

En primer lugar, En la Figura 49, se importan las bibliotecas para el funcionamiento del código, se utiliza *NumPy* para manejar y realizar cálculos matemáticos con arreglos de números (*arrays*), mientras que *pandas* se emplea para la manipulación y análisis de datos, en este caso, para leer el archivo *Excel* y gestionar los datos tabulares. La función *train_test_split* de *sklearn* permite dividir los datos en conjuntos de entrenamiento y prueba, y *StandardScaler* de *sklearn* se utiliza para estandarizar los datos (escalarlos), asegurando que cada característica tenga media 0 y desviación estándar 1, lo cual mejora el rendimiento de las redes neuronales.

```
data = pd.read_excel("/content/drive/MyDrive/simulacion_eoq.xlsx",sheet_name="Sheet1")
data
len(data)
data = data.dropna()
```

	D	K	h	L	Cantidad a pedir	Punto de reorden
0	602	53.193988	4.803646	5	115.467304	7.850101
1	935	65.869443	1.828454	2	259.549855	53.151016
2	848	61.015955	3.847163	2	164.007729	55.922705
3	770	75.765259	3.969524	4	171.445553	165.425592
4	606	55.118354	2.992482	4	149.411372	33.418055
...	...	...	...	...	...	...
9995	556	55.581834	3.269109	6	137.500443	35.989359
9996	611	72.051740	1.716667	2	226.472121	89.639394
9997	504	67.693537	2.971393	2	151.538789	98.767265
9998	909	61.165752	4.494511	9	157.293176	1.754837
9999	913	82.108971	4.093853	2	191.372515	103.647361

10000 rows × 6 columns

Figura 50. Carga de datos desde un archivo *Excel*

En la Figura 50 se visualiza la parte del código que utiliza *pd.read_excel* para leer el archivo *Excel* y selecciona la hoja "*Sheet1*", mientras que *data.dropna()* elimina cualquier fila en el conjunto de datos que contenga valores nulos para evitar problemas durante el entrenamiento del modelo.

```
[ ] X = data [["D","K","h","L"]]
    y = data["Cantidad a pedir"]
    X.shape
```

```
(10000, 4)
```

Figura 51. Selección de características (X) y el objetivo (y)

En esta parte del código, en la Figura 51, se determina X, que son las características que se utilizan para hacer la predicción, que en este caso incluyen D (Demanda), K (Costo de Pedido), h (Costo de Mantenimiento) y L (Tiempo de Entrega), mientras que y es la columna objetivo, que en este caso es la cantidad a pedir.

```
[ ]  X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.2,random_state=42)
```

Figura 52. Dividir los datos en conjuntos de entrenamiento y prueba

En esta parte del código en la Figura 52, se inicia el proceso de entrenamiento del modelo. Con la función *train_test_split* divide los datos en dos conjuntos: el conjunto de entrenamiento (*X_train, y_train*) se utiliza para entrenar la red neuronal, mientras que el conjunto de prueba (*X_test, y_test*) se emplea para evaluar el rendimiento del modelo. Se especifica *test_size*=0,2, lo que significa que el 20% de los datos se destina a pruebas y el 80% se utiliza para el entrenamiento. Además, se establece *random_state*=42 para garantizar que la división sea reproducible.

```
## Escalar los datos (X)

sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.fit_transform(X_test)
```

Figura 53. Escalado de los datos

El escalado de los datos empieza aquí en la Figura 53. *StandardScaler* se utiliza para estandarizar las características, escalando los valores de manera que tengan una media de 0 y una desviación estándar de 1. Este proceso es crucial en redes neuronales, ya que asegura que todas las características compartan el mismo rango de valores, lo que mejora el rendimiento del modelo. La función *fit_transform* ajusta el escalador a los datos de entrenamiento y, a continuación, transforma tanto los datos de entrenamiento como los de prueba

```
## Crear la red neuronal

import tensorflow as tf
```

```
[ ]  from tensorflow.keras.models import Sequential
     from tensorflow.keras.layers import Dense
```

Figura 54. Creación red neuronal

En la Figura 54, el código utiliza *TensorFlow*, una biblioteca diseñada para el aprendizaje profundo, que facilita la creación de modelos de redes neuronales. Se emplea el modelo *Sequential,* donde las capas se organizan en un flujo lineal, apilándose una encima de la otra de manera secuencial. En este modelo, se incluye

la capa *Dense*, que es una capa totalmente conectada, lo que significa que cada neurona está vinculada a todas las neuronas de la capa anterior, permitiendo una transmisión completa de la información entre las capas de la red.

```
## Red neuronal
model = Sequential()
model.add(Dense(4,activation = "relu",input_shape = (X_train.shape[1],)))
model.add(Dense(16,activation = "relu"))
model.add(Dense(16,activation = "relu"))
model.add(Dense(1))    ## si NN para predicción solo una neurona en la capa de salida
```

Figura 55. Definición de la red neuronal

En la Figura 55, el código define la estructura de la red neuronal, comienza con la capa de entrada, que toma 4 entradas correspondientes a D (Demanda), K (Costo de Pedido), h (Costo de Mantenimiento) y L (Tiempo de Entrega). Esta capa utiliza la función de activación *ReLU* (*Rectified Linear Unit*), la cual es común en redes neuronales ya que introduce no linealidad, lo que permite al modelo aprender relaciones más complejas. Luego, el modelo incluye dos capas ocultas, cada una con 16 neuronas, ambas utilizando también la activación *ReLU* para mantener la no linealidad. Finalmente, en la capa de salida, se tiene una sola neurona, ya que el objetivo es predecir un único valor: la cantidad a pedir.

```
## Compilar la red
model.compile(optimizer="adam",loss = "mse",metrics=["mae"])
```

Figura 56. Compilación del modelo

En la Figura 56, se establece las métricas y métodos de evaluación del programa. El código utiliza el optimizador *Adam* mediante la opción optimizer="*adam*", conocido por su eficiencia y robustez al trabajar con grandes cantidades de datos y redes neuronales profundas. Para evaluar el desempeño del modelo, la función de pérdida elegida es el error cuadrático medio (MSE) con *loss="mse",* que calcula la diferencia promedio entre las predicciones y los valores reales. Además, se incluye *metrics*=["*mae*"] para medir el error absoluto medio (*MAE*), proporcionando una métrica adicional para evaluar la precisión del modelo.

```
## Entrenar la red
epocas = 1000
history = model.fit(X_train,y_train,epochs = epocas,validation_split = 0.1, verbose = 10)
```

```
Epoch 1/1000
Epoch 2/1000
Epoch 3/1000
Epoch 4/1000
Epoch 5/1000
Epoch 6/1000
Epoch 7/1000
Epoch 8/1000
Epoch 9/1000
Epoch 10/1000
```

Figura 57. Entrenamiento de la red neuronal

En la Figura 57 el código empieza su entrenamiento con las redes neuronales, en primer lugar, utiliza *epochs*=1000, lo que significa que la red neuronal entrenará durante 1000 épocas, donde cada época representa un ciclo completo a través de todos los datos de entrenamiento. Además, se establece *validation_split*=0,1, lo que indica que el 10% de los datos de entrenamiento se reserva para la validación durante el entrenamiento, permitiendo evaluar el modelo a lo largo del proceso. El parámetro verbose=10 está configurado para mostrar los resultados del entrenamiento cada 10 épocas, proporcionando información periódica sobre el progreso del modelo.

```
## Evaluar el modelo
test_loss, test_mae = model.evaluate(X_test,y_test)
print(f"Error:{test_loss}, Métrica: {test_mae}")
```

```
63/63 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 14.2716 - mae: 2.5425
Error:13.323406219482422, Métrica: 2.4711172580718994
```

Figura 58. Evaluación del modelo

En la Figura 58 la función *model.evaluate* se utiliza para evaluar el rendimiento del modelo empleando el conjunto de prueba. Al ejecutar esta función, el modelo devuelve dos resultados clave: el error o pérdida (que refleja qué tan bien se ajusta el modelo a los datos de prueba) y la métrica *MAE* (Error Absoluto Medio), que mide la precisión de las predicciones en promedio. Esto proporciona una idea clara de cómo el modelo se comporta con datos no vistos durante el entrenamiento.

```
## Predicciones
predicciones = model.predict(X_test)
comparacion = pd.DataFrame(predicciones,columns = ["Predicciones"])
comparacion["Reales"] = y_test.values
comparacion
```

```
63/63 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```

	Predicciones	Reales
0	336.204865	331.642932
1	334.244629	324.859727
2	216.256821	216.104975
3	171.386948	171.732504
4	169.021378	169.055259
...	...	...
1995	135.422989	135.420480
1996	154.011978	153.647337
1997	224.910172	222.625089
1998	190.109329	189.905165
1999	215.622879	214.235450

2000 rows × 2 columns

Figura 59. Predicciones

En la Figura 59, la función *model.evaluate* se utiliza para evaluar el rendimiento del modelo empleando el conjunto de prueba. Al ejecutar esta función, el modelo devuelve dos resultados clave: el error o pérdida (que refleja qué tan bien se ajusta el modelo a los datos de prueba) y la métrica *MAE* (Error Absoluto Medio), que mide la precisión de las predicciones en promedio. Esto proporciona una idea clara de cómo el modelo se comporta con datos no vistos durante el entrenamiento.

```
y_test
```

	Cantidad a pedir
6252	331.642932
4684	324.859727
1731	216.104975
4742	171.732504
4521	169.055259
...	...
6412	135.420480
8285	153.647337
7853	222.625089
1095	189.905165
6929	214.235450

2000 rows × 1 columns

dtype: float64

```
[ ] X_test

array([[ 1.39358173,  1.29584255, -1.23259334,  1.50411605],
       [ 0.49003643,  0.57153546, -1.52040476, -0.05032503],
       [ 1.00341444,  1.72820186,  0.71398265,  1.11550578],
       ...,
       [ 0.48319139,  1.48015747,  0.14642603, -0.4389353 ],
       [ 0.64062731, -0.32025249,  0.24244234, -0.4389353 ],
       [ 1.19507557, -0.33014776, -0.16641005,  0.72689551]])
```

Figura 60. Evaluación del Modelo con Conjunto de Prueba

En la Figura 60, *X_test* y *y_test* son los conjuntos de datos utilizados para evaluar el rendimiento del modelo. *X_test* representa el conjunto de características (*features*) separadas específicamente para las pruebas; contiene valores de entrada, como la demanda (D), el costo de pedido (K), el costo de mantenimiento (h) y el tiempo de entrega (L), que no se utilizaron durante el entrenamiento del modelo. Su objetivo es verificar si el modelo puede realizar predicciones precisas con datos que no ha visto antes. Por otro lado, *y_test* corresponde a las etiquetas o resultados reales, es decir, los valores que se quieren predecir (en este caso, la cantidad a pedir). Se utiliza para comparar las predicciones generadas por el modelo con los valores reales y así evaluar su precisión.

```
nuevo = model.predict(np.array(sc.fit_transform([[784,  68.13,  0.27, 1 ]])))
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 31ms/step

nuevo
array([[194.09724]], dtype=float32)
```

Figura 61. Predicción con Nuevos Parámetros

En este último fragmento la figura 61 realiza una predicción utilizando el modelo previamente entrenado, basándose en un nuevo conjunto de características.

Primero, *np.array* crea un array de *NumPy* a partir de los valores proporcionados: D = 784, K = 68,13, h = 0,27, y L = 1. Luego, *sc.fit_transform* aplica el escalador *StandardScaler*, que se usó previamente en los datos de entrenamiento, para escalar estos valores de entrada. Este paso ajusta y transforma los datos para asegurar que sigan la misma escala que los datos originales, lo cual es esencial, ya que la red neuronal fue entrenada con datos escalados. A continuación, *model.predict* utiliza la red neuronal entrenada para realizar la predicción, tomando el array escalado como entrada. Finalmente, el resultado se almacena en la variable nuevo, que contiene la predicción de la cantidad a pedir basada en los valores introducidos ([784, 68.13, 0.27, 1]). Para finalizar se imprime el resultado de esta predicción que en este caso es 194,09724.

Esta nueva propuesta de código, basada en redes neuronales para la predicción, se ha desarrollado específicamente para el modelo clásico de cantidad económica de pedido, para el modelo de cantidad económica de pedido con discontinuidad de precios y para el modelo probabilizado de cantidad económica de pedido, para un mejor entendimiento los códigos de estos programas están en el Anexo 9. Esto se debe a que no se esta creando un nuevo programa para todos los modelos, sino que se plantea como una mejora enfocada en el programa central de esta tesis, sugiriendo cómo podría optimizarse el desempeño en estos y los demás modelos de inventarios.

4.1.5.3.1.2. Redes neuronales por clasificación

Se decidió también probar redes neuronales para clasificación con el objetivo de evaluar las mejoras que este enfoque podría ofrecer en este trabajo. En este caso, la clasificación se aplicó exclusivamente al modelo probabilístico con política S-s. Por lo tanto, la explicación que sigue se enfoca específicamente en este modelo, detallando el funcionamiento del código relacionado

```
[ ]  import numpy as np
     import matplotlib.pyplot as plt
     import tensorflow as tf
     import pandas as pd
```

Figura 62. Bibliotecas necesarias para el código

En la Figura 62, el código comienza importando varias bibliotecas esenciales: *NumPy*, que se utiliza para operaciones matemáticas y la manipulación de matrices;

matplotlib.pyplot, que sirve para generar gráficos y visualizar datos; *TensorFlow*, la biblioteca principal empleada para construir y entrenar la red neuronal; y *pandas*, utilizada para la manipulación y análisis de datos, cargando la información desde un archivo *Excel*. Estas bibliotecas proporcionan herramientas clave para la manipulación de datos, visualización y el desarrollo del modelo de clasificación con redes neuronales.

```
data = pd.read_excel("/content/drive/MyDrive/Modelos de inventarios.xlsx",sheet_name="M_política_s-S")
data
```

	x	k	c	h	p	D_max	S	s1	s2	cantidad_pedido
0	5	24.72	2.04	2.86	18.30	94	7.680	17.270	-1.910	No pedir
1	29	63.19	4.54	0.78	12.20	26	15.340	31.630	0.280	No pedir
2	21	46.50	8.61	3.81	14.29	20	6.270	16.470	-3.850	No pedir
3	6	30.32	3.95	3.08	6.46	23	6.050	18.580	-6.030	No pedir
4	22	59.36	5.29	4.88	12.96	12	5.150	14.130	-3.770	No pedir
5	30	78.71	4.27	1.76	15.90	73	48.070	74.070	22.840	Pedir
6	24	62.23	4.42	2.67	6.10	70	13.400	45.220	-18.130	No pedir
7	3	26.39	4.30	7.30	13.13	10	4.320	9.410	-0.760	No pedir
8	13	58.77	1.37	3.09	6.90	50	27.670	56.450	4.980	Pedir
9	15	26.97	8.26	3.17	13.13	63	18.820	33.330	4.410	Pedir
10	6	48.40	8.69	4.95	16.30	60	21.480	38.030	4.960	Pedir

Figura 63. Carga de datos

En la Figura 63, el código carga un archivo *Excel* que contiene datos de inventarios, como costos, demanda máxima, cantidad a pedir, entre otros, utilizando *pandas* para leer los datos desde una hoja específica. La tabla incluye variables como x, k, c, h, p, D_max, S, s1, s2, y la cantidad de pedido

```
mode_ship_label = {value: key for key,value in enumerate(data["cantidad_pedido"].unique())}
```

```
[ ] data["cantidad_pedido"] = data["cantidad_pedido"].map(mode_ship_label)
```

```
data
```

	x	k	c	h	p	D_max	S	s1	s2	cantidad_pedido
0	5	24.72	2.04	2.86	18.30	94	7.680	17.270	-1.910	0
1	29	63.19	4.54	0.78	12.20	26	15.340	31.630	0.280	0
2	21	46.50	8.61	3.81	14.29	20	6.270	16.470	-3.850	0
3	6	30.32	3.95	3.08	6.46	23	6.050	18.580	-6.030	0
4	22	59.36	5.29	4.88	12.96	12	5.150	14.130	-3.770	0
5	30	78.71	4.27	1.76	15.90	73	48.070	74.070	22.840	1
6	24	62.23	4.42	2.67	6.10	70	13.400	45.220	-18.130	0
7	3	26.39	4.30	7.30	13.13	10	4.320	9.410	-0.760	0
8	13	58.77	1.37	3.09	6.90	50	27.670	56.450	4.980	1
9	15	26.97	8.26	3.17	13.13	63	18.820	33.330	4.410	1
10	6	48.40	8.69	4.95	16.30	60	21.480	38.030	4.960	1

Figura 64. Preprocesamiento de datos

Se asignan etiquetas numéricas a la columna cantidad pedido en la Figura 64, codificando "Pedir" como 1 y "No pedir" como 0, lo que convierte la columna categórica en un formato que la red neuronal puede procesar para la clasificación.

```
X = np.asarray([data["x"],
                data["k"],
                data["c"],
                data["h"],
                data["p"],
                data["D_max"],
                data["S"],
                data["s1"],
                data["s2"]])
```

```
X = X.T
```

```
X
```

```
array([[ 5.0000e+00,  2.4720e+01,  2.0400e+00,  2.8600e+00,  1.8300e+01,
         9.4000e+01,  7.6800e+00,  1.7270e+01, -1.9100e+00],
       [ 2.9000e+01,  6.3190e+01,  4.5400e+00,  7.8000e-01,  1.2200e+01,
         2.6000e+01,  1.5340e+01,  3.1630e+01,  2.8000e-01],
       [ 2.1000e+01,  4.6500e+01,  8.6100e+00,  3.8100e+00,  1.4290e+01,
         2.0000e+01,  6.2700e+00,  1.6470e+01, -3.8500e+00],
       [ 6.0000e+00,  3.0320e+01,  3.9500e+00,  3.0800e+00,  6.4600e+00,
         2.3000e+01,  6.0500e+00,  1.8580e+01, -6.0300e+00],
       [ 2.2000e+01,  5.9360e+01,  5.2900e+00,  4.8800e+00,  1.2960e+01,
         1.2000e+01,  5.1500e+00,  1.4130e+01, -3.7700e+00],
       [ 3.0000e+01,  7.8710e+01,  4.2700e+00,  1.7600e+00,  1.5900e+01,
         7.3000e+01,  4.8070e+01,  7.4070e+01,  2.2840e+01],
       [ 2.4000e+01,  6.2230e+01,  4.4200e+00,  2.6700e+00,  6.1000e+00,
         7.0000e+01,  1.3400e+01,  4.5220e+01, -1.8130e+01],
       [ 3.0000e+00,  2.6390e+01,  4.3000e+00,  7.3000e+00,  1.3130e+01,
         1.0000e+01,  4.3200e+00,  9.4100e+00, -7.6000e-01],
       [ 1.3000e+01,  5.8770e+01,  1.3700e+00,  3.0900e+00,  6.9000e+00,
         5.0000e+01,  2.7670e+01,  5.6450e+01,  4.9800e+00],
       [ 1.5000e+01,  2.6970e+01,  8.2600e+00,  3.1700e+00,  1.3130e+01,
         6.3000e+01,  1.8820e+01,  3.3330e+01,  4.4100e+00],
```

Figura 65. Creación del arreglo X

En la Figura 65, se está creando un arreglo X a partir de varias columnas del *dataframe data*. Cada columna (x, k, c, etc.) representa una característica o atributo que se utilizará como entrada para la red neuronal. Estas características están relacionadas con el problema de inventario. Luego, se realiza una transposición (X.T) del arreglo. Esto convierte cada fila en un vector que representa los atributos de un único ejemplo, lo cual es necesario para alimentar la red neuronal.

```
y = np.asarray(data["cantidad_pedido"])
```

```
y
```

```
array([0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0,
       0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 1,
       0, 0, 0, 0, 1, 1])
```

Figura 66. Definición de y

En la Figura 66, se muestra el vector y contiene las etiquetas de los datos, que en este caso es la columna cantidad_pedido. Esta columna representa si hay que "Pedir" o

"No pedir", codificado probablemente como 1 y 0, respectivamente. Este será el valor que la red neuronal intentará predecir.

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X,y)

[ ] X_train.shape,X_test.shape, y_train.shape

((37, 9), (13, 9), (37,))
```

Figura 67. División del conjunto de datos

En la Figura 67, se utiliza la función *train_test_split* de *sklearn* para dividir el conjunto de datos en dos partes: *X_train* y *y_train,* usados para entrenar la red neuronal, y *X_test* y *y_test,* usados para evaluar el rendimiento del modelo una vez entrenado. Por defecto, se asigna el 75% de los datos para entrenamiento y el 25% para pruebas.

El siguiente comando devuelve una tupla con las dimensiones de cada conjunto de datos: *X_train.shape* muestra las dimensiones del conjunto de entrenamiento, donde la primera dimensión es el número de ejemplos y la segunda el número de características; *X_test.shape* indica las dimensiones del conjunto de prueba; y *y_train.shape* muestra las dimensiones del vector de etiquetas de entrenamiento, que normalmente es unidimensional, representando el número de ejemplos de entrenamiento.

```
##Escalar los datos

from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.fit_transform(X_test)
```

Figura 68. Normalización de los datos

En la Figura 68, se muestra la parte del código se importa la clase *StandardScaler* de la biblioteca *scikit-learn* para estandarizar las características de entrada, asegurando que los datos tengan una media de 0 y una desviación estándar de 1, lo cual es beneficioso para algoritmos de machine *learning*, especialmente redes neuronales. Se crea una instancia del escalador con *sc = StandardScaler(),* que contiene métodos para ajustar y transformar los datos. Al ejecutar *X_train = sc.fit_transform(X_train),* se ajusta el escalador a los datos de entrenamiento y se

estandariza el conjunto, calculando la media y la desviación estándar de los datos. Y se escala el conjunto de prueba con *X_test = sc.fit_transform(X_test), X_train = sc.fit_transform(X_train)*.

```
from threading import active_count
import tensorflow as tf

## input layer  9 neurons
inputlyr = tf.keras.layers.Dense(units = 9)

## hidden layers with ReLU
hdnlyr01 = tf.keras.layers.Dense(units = 16, activation = tf.nn.relu)
hdnlyr02 = tf.keras.layers.Dense(units = 16, activation = tf.nn.relu)

## output layer
outputlyr = tf.keras.layers.Dense(units = 2, activation = tf.nn.softmax)

## Crear la NN

model = tf.keras.Sequential([inputlyr,hdnlyr01,hdnlyr02,outputlyr])
```

Figura 69. Creación red neuronal

En la Figura 69, el código crea una red neuronal utilizando *TensorFlow* y *Keras*. Primero, se importa *active_count* del módulo *threading* y *tensorflow* como *tf*. Luego, se define la capa de entrada con 9 neuronas utilizando *tf.keras.layers.Dense(units*=9). A continuación, se crean dos capas ocultas, ambas con 16 neuronas y activación *ReLU* (*tf.nn.relu*), usando *tf.keras.layers.Dense*(*units*=16, *activation=tf.nn.relu*). Finalmente, se define la capa de salida con 2 neuronas y una función de activación *softmax* (*tf.nn.softmax*) para proporcionar probabilidades de clasificación. La red neuronal completa se construye en un modelo secuencial de *Keras* que incluye todas las capas en el orden definido.

```
model.compile(
    optimizer = tf.keras.optimizers.Adam(),
    loss = tf.keras.losses.SparseCategoricalCrossentropy(),
    metrics = ["accuracy"]

)
```

Figura 70. Compilación del modelo

En la Figura 70, se compila el modelo especificando tres aspectos: el optimizador *Adam*, que es eficiente para entrenar redes neuronales al ajustar los pesos y sesgos; la función de pérdida *SparseCategoricalCrossentropy*, adecuada para problemas

de clasificación con múltiples clases (en este caso, dos clases); y la precisión (*accuracy*) como métrica para evaluar el rendimiento del modelo.

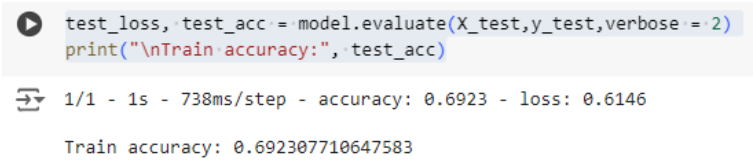


Figura 71. Evaluación del modelo

En la Figura 71, el código evalúa el rendimiento del modelo de red neuronal utilizando el conjunto de datos de prueba (*X_test* y *y_test*). Se llama a *model.evaluate()* con los datos de prueba, que calcula la pérdida (*test_loss*) y la precisión (*test_acc*) del modelo. El parámetro verbose=2 se utiliza para mostrar información detallada durante la evaluación. Después de la evaluación, se imprime la precisión de entrenamiento (*test_acc*) en la consola.

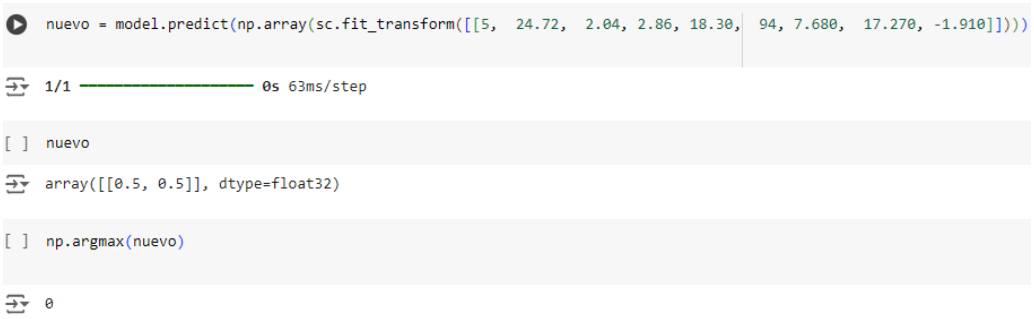


Figura 72. Nueva predicción

Luego en la Figura 72, el código nuevo = *model.predict(np.array(sc.fit_transform*([[5, 24.72, 2.04, 2.86, 18.30, 94, 7.680, 17.270, -1.910]])))) realiza una predicción utilizando el modelo, estandarizando un nuevo conjunto de datos con *StandardScaler* mediante *sc.fit_transform* antes de hacer la predicción con *model.predict*(), y almacenando el resultado en la variable nuevo. Finalmente, *np.argmax*(nuevo) se utiliza para obtener el índice del valor máximo en las predicciones, lo que permite determinar la clase predicha (la que tiene la mayor probabilidad).

```
model.summary()
```

```
Model: "sequential"
```

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 9)	90
dense_1 (Dense)	(None, 16)	160
dense_2 (Dense)	(None, 16)	272
dense_3 (Dense)	(None, 2)	34

```
Total params: 556 (2.17 KB)
Trainable params: 556 (2.17 KB)
Non-trainable params: 0 (0.00 B)
```

Figura 73. Resumen del modelo

Por último, en la Figura 73, el código *model.summary()* se utiliza para mostrar un resumen de la arquitectura del modelo de red neuronal. Este resumen incluye información sobre cada capa, como su tipo, forma de salida y el número de parámetros *entrenables*. En el resumen se observa que el modelo tiene cuatro capas: la capa de entrada (*dense*), dos capas ocultas (*dense*_1 y *dense*_2) y la capa de salida (*dense*_3). El total de parámetros *entrenables* en el modelo es 556.

Para un mejor entendimiento los códigos de este programa están en el Anexo 9.

4.1.5.4. Resultados de mejora de programa

Una vez que se han implementado las mejoras en el programa basado en redes neuronales, es fundamental llevar a cabo una comparación similar a la que se realizó con el programa de árbol de decisiones. Este análisis tiene como objetivo evaluar el grado de mejora del nuevo modelo en relación con la versión anterior. Al llevar a cabo esta comparación, se podrá determinar si ha habido una mejora significativa en la precisión de las predicciones del programa. Este proceso no solo permitirá validar la efectividad de las optimizaciones realizadas, sino que también contribuirá a una mejor comprensión de las diferencias entre los enfoques de redes neuronales y árboles de decisiones en términos de rendimiento y capacidad predictiva.

Tal como se había establecido previamente, este programa basado en redes neuronales se presenta únicamente como una propuesta de mejora al programa central de este proyecto. Por lo tanto, la comparación de resultados se limitará exclusivamente a los modelos para los cuales se ha desarrollado el código utilizando redes neuronales. Esta evaluación tiene como objetivo determinar si las modificaciones introducidas mediante la implementación de redes neuronales

logran mejoras significativas en el desempeño predictivo, en comparación con los métodos utilizados previamente, incluyendo aquellos basados en otros enfoques como los árboles de decisión y fórmulas matemáticas.

Además, este análisis permitirá identificar si la integración de redes neuronales proporciona una ventaja competitiva en términos de precisión, eficiencia, y optimización en las predicciones realizadas, justificando o no su adopción dentro del marco general del trabajo.

La Tabla 18 presenta una comparación entre tres métodos diferentes para calcular la cantidad económica de pedido (*EOQ*) y el punto de reorden en un modelo clásico de inventario.

Tabla 18. Resultados Comparativos *EOQ*: Árbol de Decisiones vs Programa vs Red neuronal

Modelo clásico de cantidad económica de pedido									
				Árbol de decisiones		**Programa**		**Red neuronal**	
D	**K**	**H**	**L**	**C. de pedido**	**P. de reorden**	**C. de pedido**	**P. de reorden**	**C. de pedido**	**P. de reorden**
784	68,13	0,27	1	315,67	29,01	629,01	154,99	421,01	175,43
659	49,38	0,07	2	263,63	71,77	964,24	353,76	380,46	176,84
1753	90,26	2,50	10	258,79	201,88	355,78	96,70	383,12	188,58
1316	96,73	2,34	10	283,83	15,56	329,85	295,88	339,75	177,95
935	44,51	2,61	1	188,90	40,39	178,58	42,11	184,86	89,01
863	81,26	2,94	11	210,05	131,27	218,42	101,10	219,39	104,13
1831	57,60	2,40	5	210,38	138,51	296,46	711,50	341,28	185,90
1483	61,12	1,39	8	289,96	194,21	361,14	307,67	391,81	180,71
1133	93,30	2,34	4	286,30	237,97	300,58	23,36	305,39	161,80
1847	16,39	0,36	3	309,04	332,187	410,10	209,74	496,39	198,85

Los métodos comparados son un enfoque basado en árboles de decisiones, un programa tradicional, y una red neuronal. Esto se hizo con el objetivo de comparar los resultados generados por los tres enfoques. Para la comparación, se realizaron 10 simulaciones en cada programa, utilizando los mismos datos como base.

Para facilitar la comprensión de la Tabla 18, se elaboraron gráficos de líneas que permiten comparar visualmente la desviación de los programas respecto a los datos exactos.

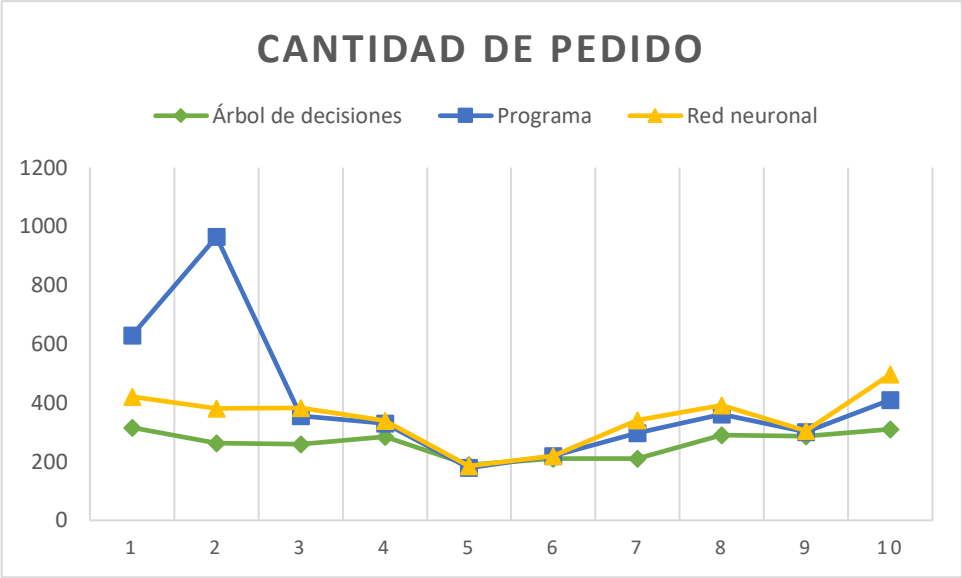


Figura 74. *EOQ* comparación, cantidad de pedido

La Figura 74 muestra los datos obtenidos de tres tipos de programas distintos. La línea azul representa los datos exactos como referencia, y visualmente se observa que la red neuronal es la que sigue de manera más fiel estos datos exactos.

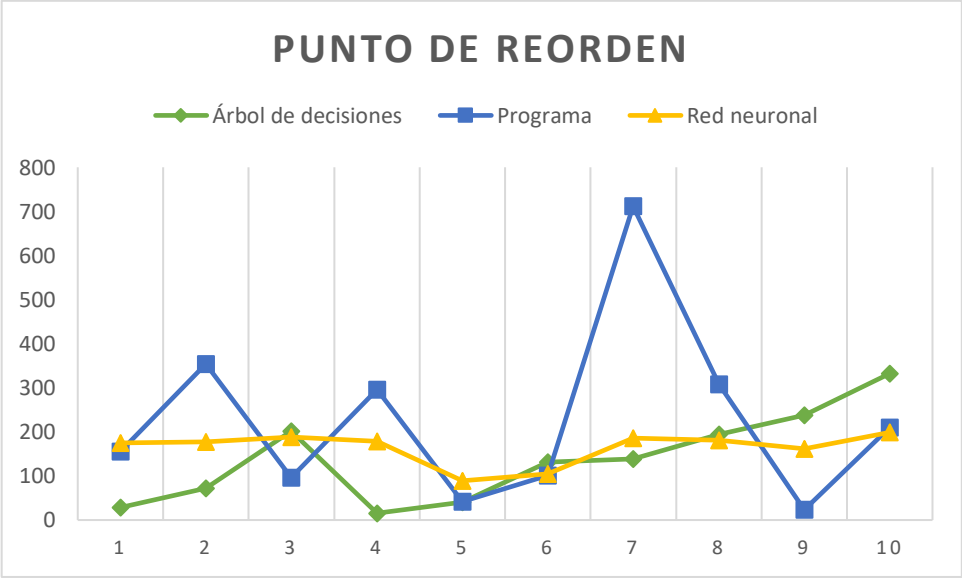


Figura 75. *EOQ* comparación, punto de reorden

La Figura 75 presenta los datos obtenidos de tres tipos de programas distintos. La línea azul sirve como referencia para los datos exactos, y visualmente se observa que ni el árbol de decisiones ni la red neuronal replican fielmente estos datos. Sin embargo, la red neuronal muestra una mayor capacidad para seguir los patrones de los datos exactos.

La Tabla 19, muestra los resultados de una comparación de error cuadrático medio.

Tabla 19. MSE con tres enfoques del modelo clásico de cantidad económica de pedido

	Comparación árbol de decisión y programa		Comparación red neuronal y programa	
	Diferencia cuadrática (cantidad de pedido)	**Diferencia cuadrática (punto de reorden)**	**Diferencia cuadrática (cantidad de pedido)**	**Diferencia cuadrática (punto de reorden)**
	98181,9556	15870,9604	43264	417,7936
	490854,3721	79518,3601	340799,0884	31300,6864
	9407,0601	11062,8324	747,4756	8441,9344
	2117,8404	78579,3024	98,01	13907,4849
	106,5024	2,9584	39,4384	2199,61
	70,0569	910,2289	0,9409	9,1809
	7409,7664	328317,5401	2008,8324	276255,36
	5066,5924	12873,1716	940,6489	16118,8416
	203,9184	46057,4521	23,1361	19165,6336
	10213,1236	14993,26781	7445,9641	118,5921
MSE	62363,11883	58818,60742	39536,75348	36793,51175
RMSE	249,73	242,53	198,84	191,82

Para este análisis se utilizaron tres enfoques diferentes para el modelo clásico de cantidad económica de pedido (*EOQ*): un árbol de la decisión, un programa con fórmulas matemáticas y una red neuronal.

La tabla 20 presenta una comparación entre tres métodos diferentes para calcular el stock de seguridad y el punto de reorden en un modelo probabilizado de cantidad económica de pedido.

Tabla 20. Resultados Comparativos modelo probabilizado: Árbol de Decisiones vs Programa vs Red neuronal

Modelo "Probabilizado" de cantidad económica de pedido									
				Árbol de decisiones		**Programa**		**Red neuronal**	
L	**D**	**σ**	**A**	**P. de reorden**	**S. de seguridad**	**P. de reorden**	**S. de seguridad**	**P. de reorden**	**S. de seguridad**
2	159	35	0,039	537,58	137,77	295,59	72,99	474,54	86,95
6	113	29	0,023	897,22	133,18	819,74	141,74	818,51	112, 34

Modelo "Probabilizado" de cantidad económica de pedido									
				Árbol de decisiones		Programa		Red neuronal	
L	D	σ	A	P. de reorden	S. de seguridad	P. de reorden	S. de seguridad	P. de reorden	S. de seguridad
7	168	36	0,049	1315,03	135,27	1333,60	157,60	1330,90	157,81
8	193	31	0,011	1717,23	160,03	1744,82	200,82	1649,71	199,48
9	155	23	0,034	1467,6	141,71	1520,93	125,93	1491,20	126,24
8	186	39	0,015	1680,33	174,24	1727,38	239,38	1757,61	236,69
2	121	27	0,044	313,9	53,4	307,14	65,14	304,95	64,42
1	174	33	0,018	235,53	59,56	243,20	69,20	415,48	66,24
6	142	25	0,027	918,27	133,18	969,99	117,99	986,23	117,95
1	108	38	0,041	150,01	84,9	174,09	66,09	172,71	65,88

Los métodos comparados son un enfoque basado en árboles de decisiones, un programa tradicional, y una red neuronal. Esto se hizo con el objetivo de comparar los resultados generados por los tres enfoques. Para la comparación, se realizaron 10 simulaciones en cada programa, utilizando los mismos datos como base.

Para facilitar la comprensión de la Tabla 20, se elaboraron gráficos de líneas que permiten comparar visualmente la desviación de los programas respecto a los datos exactos.

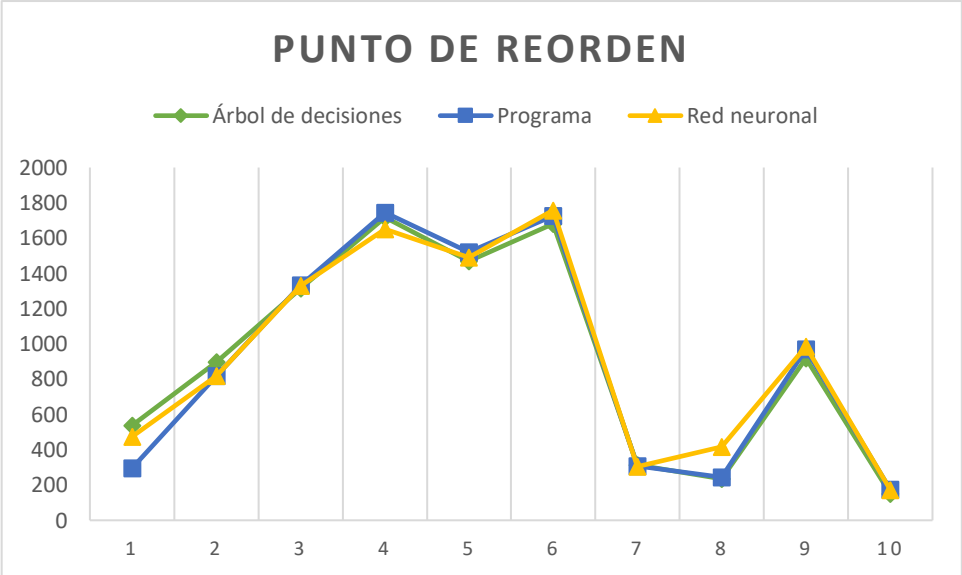


Figura 76. EOQ Probabilizado, comparación punto de reorden

La Figura 76 presenta los datos obtenidos de tres tipos de programas distintos. La línea azul representa los datos exactos como referencia y, visualmente, se observa que

tanto el árbol de decisiones como la red neuronal siguen los datos exactos con alta precisión.

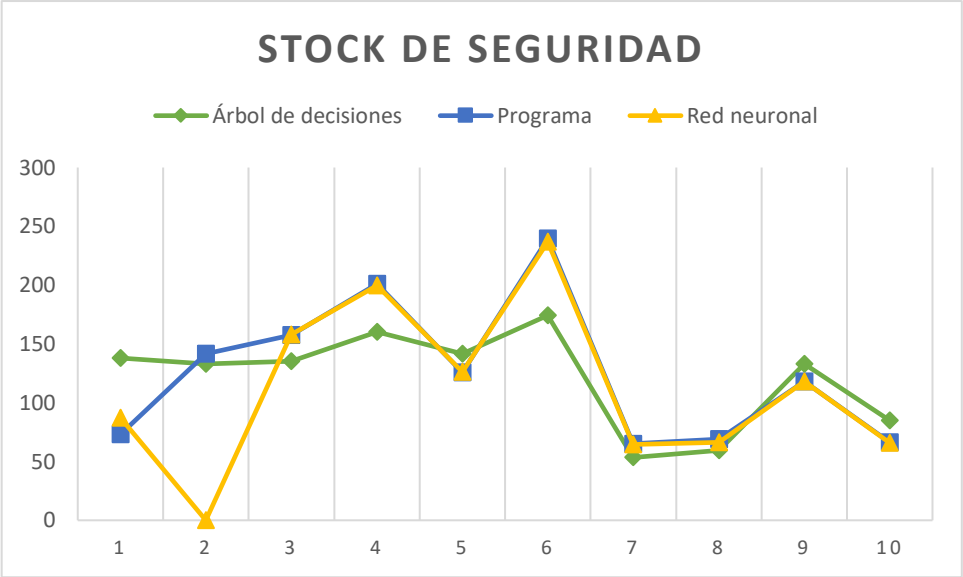


Figura 77. EOQ Probabilizado, comparación stock de seguridad

La Figura 75 muestra los datos obtenidos de tres programas distintos. La línea azul representa los datos exactos como referencia y, visualmente, se observa que, salvo en los dos primeros datos, la red neuronal sigue los datos exactos con una precisión cercana al 100%, superando notablemente al árbol de decisiones.

La tabla 21, presenta una comparación de los errores de cuadráticos medios (MSE).

Tabla 21. MSE con tres enfoques del Modelo Probabilizado

	Comparación árbol de decisión y programa		**Comparación red neuronal y programa**	
	Diferencia cuadrática (punto de reorden)	**Diferencia cuadrática (stock de seguridad)**	**Diferencia cuadrática (punto de reorden)**	**Diferencia cuadrática (stock de seguridad)**
	58559,1601	4196,4484	32023,1025	194,8816
	6003,1504	73,2736	1,5129	864,36
	344,8449	498,6289	7,29	0,0441
	761,2081	1663,8241	9045,9121	1,7956
	2844,0889	249,0084	883,8729	0,0961
	2213,7025	4243,2196	913,8529	7,2361
	45,6976	137,8276	4,7961	0,5184
	58,8289	92,9296	29680,3984	8,7616
	2674,9584	230,7361	263,7376	0,0016
	579,8464	353,8161	1,9044	0,0441
MSE	7408,54862	1173,97124	7282,63798	107,77392

	Comparación árbol de decisión y programa		Comparación red neuronal y programa	
	Diferencia cuadrática (punto de reorden)	**Diferencia cuadrática (stock de seguridad)**	**Diferencia cuadrática (punto de reorden)**	**Diferencia cuadrática (stock de seguridad)**
RMSE	86,07	34,26	85,35	10,38

Se analizo el modelo probabilístico de cantidad económica de pedido utilizando tres enfoques: un árbol de la decisión, un programa con fórmulas matemáticas, y una red neuronal.

4.1.5.4.1. Resultado de simulaciones de los 3 programas

Al finalizar el análisis, se refleja que, en el modelo clásico de cantidad económica de pedido, el enfoque basado en redes neuronales genera un menor error cuadrático medio tanto en la cantidad de pedido como en el punto de reorden en comparación con el enfoque basado en árboles de decisión. Esto indica que las redes neuronales ofrecen una mejora significativa en la precisión de las predicciones, siendo una opción más prometedora para mejorar la exactitud en el modelo clásico de cantidad económica de pedido.

Además, en la tabla que compara el árbol de decisión y la red neuronal frente a un programa de programación en el modelo probabilístico, la red neuronal ofrece mejores predicciones, especialmente en el *stock* de seguridad, donde el error cuadrático medio es mucho más bajo en comparación con el árbol de decisión. Aunque para el punto de reorden ambos enfoques presentan errores significativos, la red neuronal sigue mostrando una ventaja leve en términos de consistencia y precisión.

Como conclusión, el árbol de decisión parece ser menos confiable en la predicción, se evidencia una clara ventaja a favor de la red neuronal. Esto sugiere que las redes neuronales podrían ser la opción más prometedora para estos escenarios.

4.1.6. Prueba de hipótesis

Para realizar la prueba de hipótesis en este estudio, se utilizó una prueba t de *Student* para evaluar si existe una diferencia significativa entre las dos muestras de datos, con el objetivo de decidir si la hipótesis se rechaza o se acepta. Se eligió un nivel de

significancia de p=0.05 para este análisis. Idealmente, el análisis debería basarse en al menos 100 datos para cada grupo: uno de los datos antes de la implementación del programa y otro después de la implementación.

Sin embargo, solo se cuenta con 10 datos empíricos de empresas que proporcionaron información limitada, lo que dificulta realizar una comparación estadística robusta. Para abordar esta limitación, se empleó el método de muestreo *bootstrap*, que permite generar datos adicionales basados en la muestra original. Así, se expandió la muestra a 100 datos, creando una distribución más amplia y representativa de valores. Este proceso proporciona una base estadística más sólida para la comparación entre las prácticas reales de las empresas y los resultados teóricos del programa.

Para realizar esta prueba de hipótesis, el análisis se enfocó exclusivamente en el programa de árbol de decisiones, utilizando el modelo *EOQ* (Cantidad Económica de Pedido) y la variable "cantidad a pedir". La razón de esta elección es que, si este programa presenta una diferencia significativa en comparación con los datos empíricos, se puede inferir que el programa de redes neuronales también la tendría.

Esto se debe a que los resultados previos han mostrado que las redes neuronales ofrecen una mejor precisión en la predicción de datos. Si se encuentra una diferencia significativa en el análisis de la cantidad a pedir utilizando el modelo *EOQ* y el programa de árbol de decisiones, se puede asumir con confianza que el programa basado en redes neuronales ofrecería resultados aún más alineados con los valores empíricos, validando así la hipótesis sobre la precisión del programa.

En primer lugar, se recopilaron las cantidades a pedir de las diferentes empresas a las que se realizaron entrevistas. Con base en estos datos, se aplicó el método de muestreo *bootstrap* para aumentar el tamaño de la muestra. Posteriormente, se utilizó un programa para generar 100 datos de cantidad a pedir, lo que permitió construir una tabla comparativa con ambos tipos de datos. Para una comprensión más detallada del proceso de cálculo del muestreo *bootstrap* y la elaboración de la tabla, se puede consultar el Anexo 10.

Con estos datos, se utilizó un código para verificar, en primer lugar, la normalidad de estos y, posteriormente, realizar la prueba t de *Student*. El código utilizado se encuentra en el Anexo 11.

Al aplicar el código, se generó la figura 78.

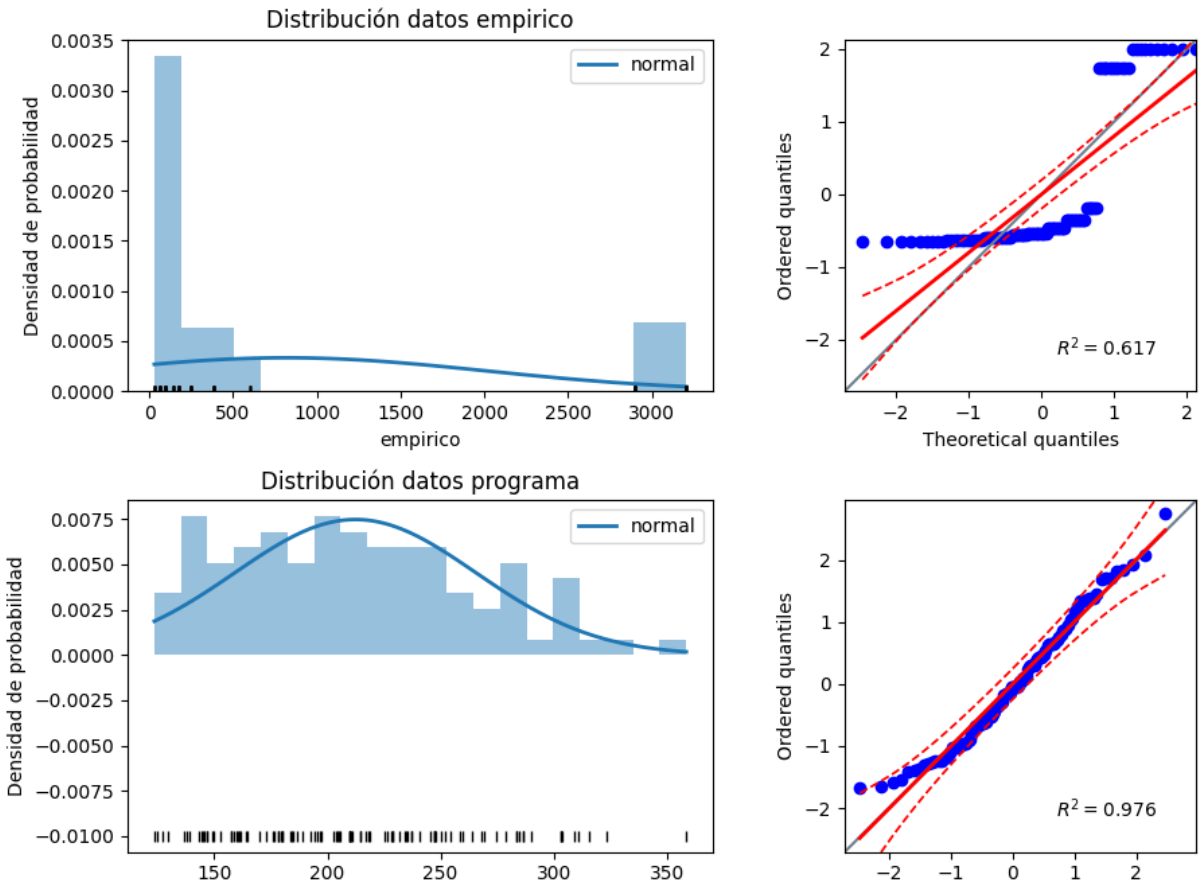


Figura 78. Normalidad de los datos

Con una observación rápida, se puede inferir que los datos empíricos no siguen una distribución normal, ya que presentan una distribución asimétrica. En el gráfico de cuantiles (Q-Q), los puntos no siguen la línea de referencia recta, especialmente en los extremos, lo cual confirma la desviación de la normalidad. Además, el valor de R2=0,617 indica una baja correlación con la normalidad.

Por otro lado, los datos generados por el programa muestran una distribución más simétrica y ajustada a una curva normal. En el gráfico de cuantiles, los puntos siguen de manera más cercana la línea de referencia, sugiriendo un ajuste adecuado a la normalidad. El valor de R2=0,976 respalda una alta correlación con la normalidad, lo cual apoya la hipótesis de que estos datos pueden considerarse normalmente distribuidos.

A continuación, en el mismo código, se realizó la prueba de normalidad *Shapiro-Wilk* para verificar la normalidad de los datos, obteniendo el valor p. Los resultados de esta prueba se muestran en la Tabla 22.

Tabla 22. Valor p normalidad

Test de *normal dad Shapiro-Wilk*			Normalidad
	Datos empíricos	**Datos programas**	Falso
valor p	6,848007e-15	0,031024	Falso

La Tabla 22, presenta los resultados de la prueba de normalidad *Shapiro-Wilk* para dos conjuntos de datos: "Datos empíricos" y "Datos del programa".

Para los Datos empíricos, el valor p es 6,848007e-15, que es extremadamente pequeño, mucho menor que el valor p utilizado de 0,05. Esto indica que los datos no siguen una distribución normal.

Para los Datos del programa, el valor p es 0,031024, que es menor que 0,05, lo que también indica que los datos no siguen una distribución normal.

Ambos conjuntos de datos muestran resultados que rechazan la hipótesis de normalidad, lo que implica que ninguno de los dos sigue una distribución normal. Sin embargo, dado que el tamaño de cada grupo es mayor que 30 se puede considerar que el t-test puede seguir siendo válido, aunque los datos no sigan perfectamente una distribución normal, debido a que el teorema central del límite asegura que la distribución de la media de una muestra grande tiende a ser normal, independientemente de la distribución de los datos subyacentes.

T-TEST

```
[11] # Test para datos independientes (p-value, intervalos de confianza)
     # ============================================================================

     pg.ttest(x=empirico, y=programa, correction=False)
```

	T	dof	alternative	p-val	CI95%	cohen-d	BF10	power
T-test	5.031272	198	two-sided	0.000001	[368.24, 842.99]	0.711529	1.277e+04	0.998843

Figura 79. Resultados de la prueba t para datos independientes

Una vez verificada la normalidad, se aplicó la prueba t, obteniendo un p-valor de 0,00001, como se muestra en la Figura 79. Este resultado indica que hay suficiente evidencia para afirmar que existe una diferencia significativa entre las cantidades observadas y las calculadas por el modelo *EOQ* basado en árboles de decisión. Por lo tanto, se rechaza la hipótesis nula. Además, el tamaña del efecto medido por d-

cohen es alto (0,711529), lo que indica que existe una diferencia neta importante entre las dos muestras de datos.

4.1.6.1. Optimización de recursos

Después de realizar la prueba de hipótesis, se determinó que existe una diferencia significativa entre los dos conjuntos de datos, lo que llevó al rechazo de la hipótesis nula. Sin embargo, es importante explicar cómo este resultado también respalda la optimización de recursos en las empresas que implementan este programa en la gestión de inventarios.

El primer objetivo de este trabajo demostró que las empresas que gestionan sus inventarios sin apoyo tecnológico suelen enfrentar problemas de faltantes, sobrantes o ambos, debido a que realizan sus aprovisionamientos de manera empírica. Aunque la amplia experiencia de algunas empresas les permite reducir el impacto de estos errores, la implementación del programa puede minimizar significativamente estos problemas y mejorar la eficiencia en la gestión de inventarios. En la tabla 23, se evidencias los problemas que presentaron las empresas.

Tabla 23. Problemas identificados en las empresas analizadas.

Empresa	Problemática
A TODP PEDAL	Sobrantes
Abastos el Nazareno	Faltantes
Águila Bike	Sobrantes y Faltantes
Comercial Enríquez	Sobrantes
Ferroalvid	Sobrantes y Faltantes
Mega Ofertas	Sobrantes
Movitech	Sobrantes
SALOMON	Sobrantes
Technet	Sobrantes
Tejidos Olarte	Sobrantes

Como se muestra en la tabla 23, todas las empresas entrevistadas, sin excepción, enfrentan problemas de faltantes o sobrantes, ya sean grandes o pequeños. En este

punto, se evidencia y justifica la optimización de recursos, ya que la prueba de hipótesis se realizó con los datos recopilados de estas empresas, y los resultados mostraron una diferencia significativa entre sus datos y los generados por el programa. Los valores obtenidos a través del programa son más cercanos a una gestión óptima de inventarios, ya que siguen un enfoque más preciso basado en modelos óptimos de gestión. Dado que la diferencia entre ambos conjuntos de datos es considerable, la implementación de este sistema permitiría a las empresas reducir significativamente los errores derivados de una gestión empírica, optimizando costos, tiempo y recursos en el proceso.

Del mismo modo, la optimización de recursos en las empresas se logra al determinar con precisión la cantidad de pedido y el punto de reorden. Un problema evidente era el tiempo dedicado a realizar pedidos, especialmente cuando se trabajaba con múltiples proveedores. Sin embargo, al contar con estos dos parámetros, se reduciría significativamente el tiempo empleado en esta tarea, ya que los pedidos se realizarían únicamente cuando el inventario alcanzara el punto de reorden calculado. De este modo, se eliminaría la necesidad de destinar tiempo y recursos adicionales a esta gestión, mejorando la eficiencia operativa.

Para realizar un ejemplo práctico de la optimización de recursos y costos que nos puede ofrecer el programa, se seleccionó una de las empresas analizadas. En la Tabla 24 se presentan los valores de compras y ventas de la empresa Águila Bike.

Tabla 24. Información empresa Águila Bike.

Productos	Compra		Venta		Precio
	Cantidad mensual	T. promedio de reabastecimiento	Cantidad mensual	Periodo entre compra y venta	
Llantas de bicicleta	1000	Cada mes	1000	1 mes y medio	15
Tubos de bicicleta	1000	Cade mes	1000	1 mes y medio	3
Equipamiento de ciclismo	100	Dos veces al año	100	1 año	25
Repuestos y Accesorios de moto	1000	Cada semana	1000	2 meses	5
Equipamiento de moto	100	Cuatro veces en el año	100	6 meses	50

Los datos proporcionados en la Tabla 24, que detallan las ventas y compras de la empresa Águila Bike, fueron la base para desarrollar el ejemplo. A través de las entrevistas, se obtuvo información sobre los principales productos que comercializa la empresa, la cantidad de producto que solicita en cada período, el tiempo en que se vende lo adquirido y el precio unitario de cada artículo. En este caso, la empresa maneja productos relacionados con bicicletas y motocicletas.

Se identificó que todas las empresas utilizan el modelo de inventario EOQ y que ninguna implementa un sistema de inventario complejo, sino que optan por el modelo más básico. Con esta información, se emplearon los datos recopilados para calcular los parámetros del EOQ para cada producto, con el fin de analizar la diferencia y optimización que ofrece el programa al comparar los sobrantes y costos antes y después de su implementación.

En la tabla 25 se muestran los diferentes parámetros del EOQ por cada producto de la empresa.

Tabla 25. Cálculo de Cantidad Óptima de Pedido y Punto de Reorden.

Productos	D	K	H	L	Cantidad a pedir	Punto de reorden
Llantas de bicicleta	666,66	70	0,2	0,033	683	22
Tubos de bicicleta	666,66	70	0,2	0,033	683	22
Equipamiento de ciclismo	100	12	0,2	0,0028	107	10
Repuestos y accesorios de moto	500	25	0,2	0,167	547	83
Equipamiento de moto	200	14	0,2	0,167	223	33

La Tabla 25 presenta los parámetros obtenidos a partir de los datos de la Tabla 24. Estos valores se calcularon utilizando la información disponible, en el caso de la demanda de llantas para bicicleta según los datos recopilados, la empresa adquiere 1,000 unidades mensuales, y estas se agotan en un mes y medio. Aplicando una regla de tres simple, se obtiene que la demanda mensual es de 666.66 unidades. Este mismo procedimiento se aplicó para cada producto. Para el costo de pedido, se realizó una estimación con base en la información obtenida en las entrevistas, mientras que el costo de almacenamiento y el tiempo de entrega también fueron determinados a partir de dichas entrevistas. Cabe destacar que estos datos están determinados según las unidades de tiempo proporcionadas por la empresa. Por ejemplo, la

demanda de llantas para bicicleta se calculó en meses, mientras que la de los equipos de ciclismo se estimó en años.

Con estos datos, se utilizó el programa EOQ basado en redes neuronales para calcular la cantidad óptima de pedido y el punto de reorden. A partir de estos resultados, se realizó un análisis para determinar si el programa proporciona una optimización en comparación con los datos reales. En la tabla 26 se muestra la comparación de lo real con los datos obtenidos con el programa.

Tabla 26. Optimización: Comparación Real vs. Programa

Productos	**Excedentes**		**Costos**	
	e1	**e2**		**Programa**
Llantas de bicicleta	333,34	16,34	5000,1	245,1
Tubos de bicicleta	333,34	16,34	1000,02	49,02
Equipamiento de ciclismo	100	7	2500	175
Repuestos y accesorios de moto	3500	47	17500	235
Equipamiento de moto	200	23	10000	1150
			36000.12	1854.12
		Optimización		94.84

La Tabla 26 muestra la optimización lograda con el programa. La columna e1 representa el excedente real de unidades que tiene cada producto por mes o año, según corresponda, calculado como la diferencia entre las compras y las ventas. Se puede observar que la empresa mantiene un alto nivel de sobrantes en todos sus productos.

Por otro lado, la columna e2 muestra los excedentes que se generarían al aplicar la cantidad óptima de pedido calculada por el programa. Para obtener estos valores, se restó la demanda o ventas de cada producto a la cantidad recomendada por el programa. Como resultado, los sobrantes se reducen significativamente en comparación con los valores actuales. Por ejemplo, en el caso de las llantas para bicicleta, el excedente real de 333.34 unidades se reduciría a solo 16.34 unidades al

implementar el modelo, lo que demuestra una mejora notable. De manera similar, todos los demás productos presentan una optimización comparable.

La columna de Costos muestra el impacto financiero que tiene para la empresa el exceso de inventario. En la columna "Real", se presenta el costo asociado a la compra de las unidades sobrantes. Por ejemplo, en el caso de las llantas para bicicleta, el excedente de 333.34 unidades genera un costo adicional de 5,000.10 dólares por mes. Por otro lado, la columna "Programa" indica el costo de los sobrantes al aplicar la cantidad óptima de pedido calculada por el programa. Siguiendo esta metodología, el costo adicional de las llantas para bicicleta se reduciría a 245.10 dólares mensuales. Esto representa una optimización significativa en costos e inventario.

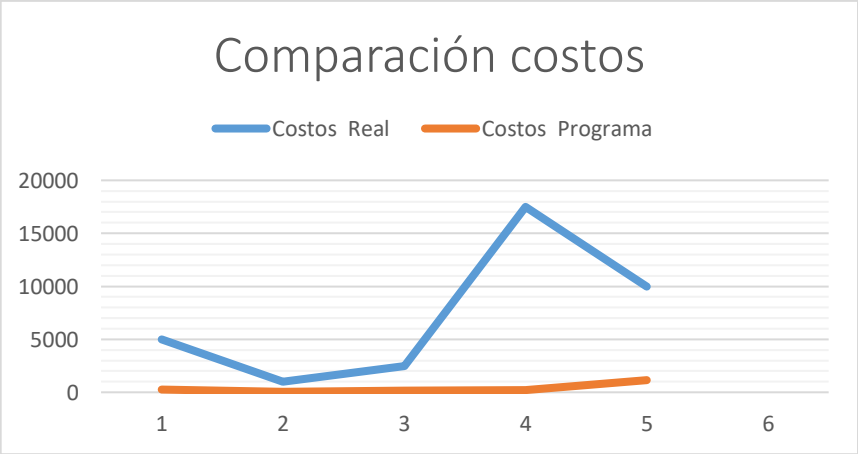


Figura 80. Comparación costos real vs programa

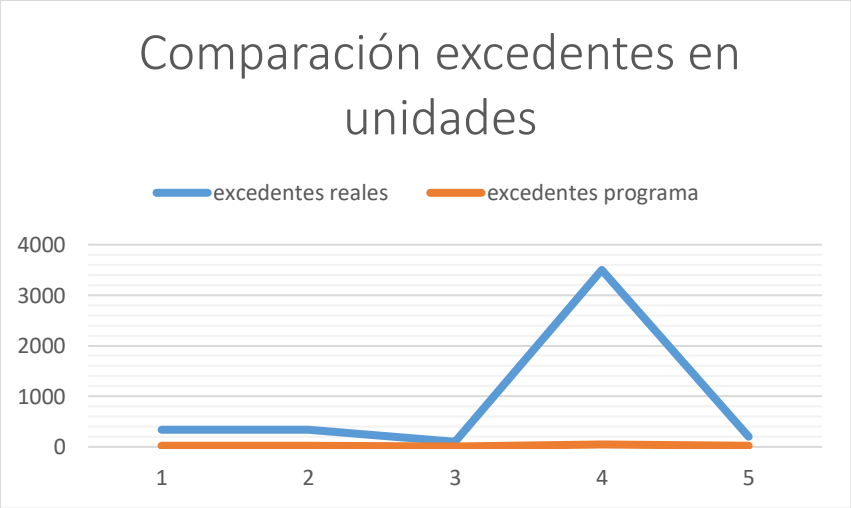


Figura 81. Comparaciones excedentes real vs programa

En las figuras 80 y 81 se compara la diferencia entre los datos reales de la empresa y los resultados obtenidos tras la optimización del programa. Se observa que todos los productos siguen un mismo patrón de reducción, reflejando una mejora en la eficiencia. Esto demuestra el impacto positivo que el programa puede generar en la gestión de inventarios.

Considerando que el costo adicional total que la empresa incurre actualmente por todos los productos analizados es de 36,000.12 dólares, y al utilizar el programa el costo total se reduce a 1,854.12 dólares, se empleó la siguiente fórmula para calcular el porcentaje de optimización que se podría lograr aplicando el programa a la gestión de inventarios de la empresa.

$$\boldsymbol{Optimización} = \left(\frac{\boldsymbol{Costo\ antes - Costo\ ahora}}{\boldsymbol{Costo\ antes}}\right) \times \mathbf{100} \qquad (46)$$

Al aplicar esta fórmula, se observa que es posible lograr una optimización de costos e inventarios superior al 94%, lo cual constituye una mejora significativa que se puede alcanzar simplemente implementando el programa. Además, este 94% de optimización está en línea con el 5% de margen de error del programa utilizado.

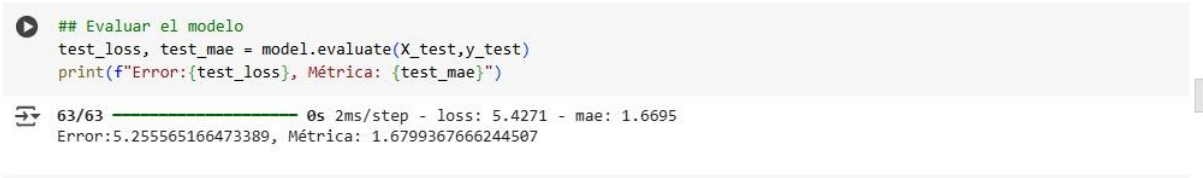


Figura 82. Error EOQ redes neuronales

En la figura 80 se presenta la métrica de error obtenida a partir del programa de redes neuronales del modelo EOQ.

4.2. DISCUSIÓN

A lo largo de este trabajo de titulación, se tomaron como referencia diversas investigaciones que contribuyeron al desarrollo adecuado de este proyecto. Los distintos enfoques, resultados y metodologías empleadas en estos estudios fueron

fundamentales para orientar y enriquecer el proceso de investigación, lo que permitió que este trabajo se llevara a cabo de manera efectiva y coherente. La integración de estas referencias no solo proporcionó un marco teórico sólido, sino que también facilitó la identificación de mejores prácticas y estrategias que se tradujeron en un resultado final mejor de este trabajo.

En la investigación desarrollada por Aguilar (2021), se desarrolló un software específicamente diseñado para que la empresa TUS ENKANTOS optimizara sus procesos de inventario y compras. Este programa se basó en el modelo de Cantidad Económica de Pedido (*EOQ*) y la revisión continua, lo que contribuyó significativamente a la mejora de sus procesos de gestión logística. Este trabajo fue esencial para el desarrollo de esta tesis, ya que proporcionó una referencia valiosa sobre cómo el uso de tecnologías avanzadas y modelos de inventario puede optimizar las operaciones de una empresa.

La investigación de Aguilar implementó programas para la gestión de inventarios, demostrando cómo la adopción de este tipo de tecnología puede influir positivamente en la eficiencia y efectividad operativa. A diferencia de esa tesis, que se centró exclusivamente en el modelo *EOQ*, esta investigación tiene un enfoque más amplio al incluir todos los modelos de inventario existentes. Este trabajo sirvió como un importante ejemplo de cómo podría llevarse a cabo la implementación en una empresa que optara por el modelo *EOQ*. Esto proporciona evidencia sólida de las ventajas de implementar los programas desarrollados en esta tesis.

Además, los programas realizados en esta investigación, al abarcar múltiples modelos, permiten una adaptación a las necesidades específicas de diferentes organizaciones. De esta manera, esta investigación se enriquece con las lecciones aprendidas de la obra de Aguilar, integrando su enfoque práctico para abordar los desafíos de gestión de inventarios en diversas empresas.

El trabajo elaborado por Chávez (2021), se centra en la implementación y adopción de modelos de inventario en una empresa. Aunque este estudio no se dedica a la creación de un sistema tecnológico, que es el enfoque principal de esta tesis, proporciona una evidencia valiosa sobre cómo el uso de modelos de inventario puede generar una considerable optimización en la gestión empresarial.

En su investigación, Chávez identifica las deficiencias que surgen al gestionar los procesos logísticos de manera empírica, destacando los errores significativos que

pueden ocurrir cuando no se utilizan modelos adecuados para la empresa. Su trabajo aborda específicamente la implementación del modelo de Cantidad Económica de Pedido (*EOQ*) y el método ABC, esta implementación permitió a la empresa mejorar sus procesos logísticos y optimizar el 63% de su inventario, lo que coinciden con los hallazgos de esta investigación donde también destacan los beneficios de estos modelos en la gestión de inventarios y en la reducción de ineficiencias. Este análisis refuerza la idea de que incluso la implementación de modelos de inventario relativamente simples puede resultar en beneficios sustanciales para las empresas.

Además, el programa desarrollado en esta tesis incluye una amplia variedad de modelos de inventario, teniendo así el potencial de proporcionar optimizaciones y beneficios similares a una gama más amplia de organizaciones. La investigación de Chávez sirve como referencia clave para entender cómo podría aplicarse este programa en una empresa real, mostrando que la integración de modelos de inventario adecuados no solo mejora la eficiencia, sino que también puede transformar la manera en que las empresas gestionan sus recursos logísticos. Este trabajo destaca la importancia de incorporar modelos estructurados en la gestión de inventarios.

El trabajo realizo por Valera (2023), se enfoca en la optimización de los recursos mejorando toda la gestión logística de la empresa Grupo ABANSAC, este trabajo en comparación busca de igual forma la optimización de recursos mediante el programa creado. Se observa que ambos estudios coinciden en la importancia de la gestión de inventarios como un factor determinante para la optimización de los recursos en las empresas. Al igual que en esta tesis, la investigación destaca la importancia de las empresas que no implementan sistemas tecnológicos avanzados retrasos y presenta deficiencias en sus procesos logísticos, afectando su eficiencia operativa. Ambos estudios concluyen que la implementación de herramientas tecnológicas es esencial para mejorar estos aspectos, lo que en el presente trabajo se logra el desarrollo de programas específicos en fórmulas matemáticas, árboles de decisión y redes neuronales, los cuales optimizan tanto la gestión como la toma de decisiones con relación a inventarios.

Otro aspecto relevante en el que coinciden todos los estudios es la necesidad de adaptar los modelos de gestión de inventarios a las características específicas de cada organización. Este trabajo demuestra, a través de sus resultados, una mejora de

más del 30% en todos los procesos analizados, gracias a la implementación de mejoras en la gestión logística. Al igual que en investigaciones anteriores, este estudio resalta los beneficios de optimizar la gestión logística, en este caso particular, a través de la mejora en la gestión de inventarios. Estas mejoras no solo incrementan la eficiencia operativa, sino que también contribuyen a una gestión más efectiva de los recursos, lo que resulta en un impacto positivo en el rendimiento general de la organización.

V. CONCLUSIONES Y RECOMENDACIONES

5.1. CONCLUSIONES

1- La gestión de inventarios en empresas que operan sin sistemas tecnológicos revela enfoques relacionados con las compras, el almacenamiento y las ventas. A través de entrevistas y análisis documental, se identificó que la mayoría de estas empresas no emplea tecnología, lo que genera retrasos en los procesos. El estudio destaca la necesidad de implementar herramientas tecnológicas para optimizar estas actividades, especialmente en sectores donde el tiempo es un factor importante. El flujograma desarrollado proporciona una base sólida para comprender y mejorar los procesos logísticos de estas empresas.

2 - El análisis de los diferentes modelos de gestión de inventarios ha demostrado que no existe un modelo universal que se pueda aplicar a todas las empresas. Cada organización tiene características particulares que dependen de su tamaño, sector económico y demanda, lo que exige una selección cuidadosa del modelo de inventario más adecuado.

Los modelos detallados en la Tabla 7 facilitan la personalización y ajuste del sistema de inventarios, lo que permite optimizar recursos, reducir costos y mejorar la eficiencia operativa, factores clave para el éxito en la gestión de inventarios.

3 - La elección de *Python* como lenguaje para el desarrollo del sistema de gestión de inventarios basado en un árbol resulto ser la opción correcta, debido a su facilidad de uso, su amplia gama de bibliotecas especializadas y su capacidad de integración con otros sistemas. *Python* no solo facilitó el desarrollo y mantenimiento del software, sino que también proporcionó las herramientas necesarias tanto para la creación del árbol de decisiones como de la red neuronal, convirtiéndose en un aliado clave para el éxito de este proyecto.

En total, se desarrollaron tres tipos de programas distintos, todos con el objetivo: optimizar la gestión de inventarios, uno de fórmulas matemáticas exactas, árboles de decisión y redes neuronales. La creación de estos programas se basó en las fórmulas y modelos presentados en el libro Investigación Taha (2004), que sirvió como referencia principal para todos los cálculos y métodos aplicados.

4 - El primer programa, basado en fórmulas matemáticas, fue diseñado para abordar todos los modelos de inventarios estudiados, proporcionando un enfoque directo y preciso para la resolución de problemas específicos. El segundo, basado en un árbol de decisiones, ofreció una estructura lógica que permite la toma de decisiones en función de los datos y escenarios identificados, aplicándose también a todos los modelos analizados.

Finalmente, se desarrolló un tercer programa utilizando redes neuronales, concebido como una mejora del modelo de árbol de decisiones. Sin embargo, este enfoque se limitó a modelos específicos, donde se buscó una mayor capacidad de adaptación y predicción en escenarios más complejos, aunque no se implementó para la totalidad de los modelos.

En conjunto, estos programas representan un avance significativo en la gestión de inventarios, ya que permiten no solo la automatización y optimización de procesos, sino también la posibilidad de mejorar la toma de decisiones con el uso de tecnologías avanzadas como las redes neuronales. Este trabajo refleja cómo el enfoque matemático tradicional puede complementarse con innovaciones tecnológicas para ofrecer soluciones más eficientes y precisas.

5 - Las simulaciones realizadas con todos los programas desarrollados arrojaron resultados muy variados. En primer lugar, como era de esperarse, el programa basado en fórmulas matemáticas proporcionó resultados exactos, los cuales sirvieron como referencia para comparar el rendimiento de los otros dos modelos. El análisis mostró que, aunque el programa basado en el árbol de decisiones logró acercarse a los datos reales, presentó una desviación significativa en sus predicciones. Por otro lado, el modelo de redes neuronales también mostró algunas desviaciones, pero estas fueron considerablemente menores en comparación con las del árbol de decisiones.

Esto evidencia que, si bien ambos enfoques tienen margen de mejora, el modelo de redes neuronales ofrece una mayor precisión y consistencia en sus resultados.

En este contexto, se puede concluir que la propuesta de mejora fue todo un éxito. La implementación de redes neuronales no solo redujo las imprecisiones en las predicciones, sino que también demostró ser una herramienta valiosa para optimizar la gestión de inventarios en escenarios donde los datos no son perfectamente lineales o predecibles. Este avance no solo valida la viabilidad del enfoque, sino que también abre la puerta a futuras mejoras y optimizaciones en los procesos empresariales, asegurando un sistema más robusto y adaptable a las necesidades cambiantes del entorno empresarial. De igual manera, se demuestra la optimización de recursos de forma notable, alcanzando una mejora del 94% en el ejemplo de implementación del programa.

5.2. RECOMENDACIÓN

1 - El uso del programa basado en árboles de decisiones y redes neuronales depende fundamentalmente de la calidad de la base de datos que se utilice. Por ello, es esencial emplear el primer programa basado en fórmulas matemáticas para complementar ambos enfoques. Se recomienda que, al aplicar estos programas en una empresa, primero se genere una base de datos utilizando el programa matemático, y luego se cargue esa base en los otros dos programas. Dado que ambos modelos trabajan con predicciones, su rendimiento mejorará considerablemente al usar datos reales y específicos de la empresa.

Actualmente, el sistema cuenta con una base de datos cuyos valores están muy dispersos, dado que se generaron aleatoriamente sin tener en mente ningún comportamiento especifico, lo que afecta la precisión de las predicciones. Sin embargo, al utilizar datos directamente obtenidos de la empresa, será posible identificar patrones y delimitar con mayor exactitud las características propias de la operación, optimizando así la predicción. En resumen, al trabajar con información concreta y precisa de la empresa, el rendimiento de los modelos será significativamente más alto, ya que se ajustarán mejor a los patrones y necesidades reales del negocio

2 - En el entrenamiento de redes neuronales, se recomienda utilizar siempre 1000 ciclos de entrenamiento. Aunque este número puede incrementar el tiempo de compilación, se observó que proporciona una mejor aproximación a los datos reales en todos los modelos en los que se implementaron redes neuronales

Los programas desarrollados están diseñados específicamente para calcular la cantidad de pedido y el punto de reorden, como se explicó en secciones anteriores de este trabajo. Este enfoque se eligió debido a la limitación que existía en otros programas para realizar estas funciones en modelos de inventario más específicos. Por lo tanto, se recomienda complementar este software con una aplicación básica de gestión de inventarios que facilite el registro de ventas, inventarios y compras. Esto mejorará la eficacia al utilizar el programa.

VI. REFERENCIAS BIBLIOGRÁFICAS

Aguilar, J. (2021). *Diseño de un modelo de gestión en base tecnológica para la optimización de procesos de pedidos a proveedores de los negocios comerciales en la ciudad de Tulcán.* Tulcán: Universidad Politécnica Estatal del Carchi. Obtenido de http://repositorio.upec.edu.ec/handle/123456789/1459

Arora, S., & Barak, B. (2009). *Complexity theory: A modern approach.* Cambridge: Cambridge University Press.

Bayer, M. (2022). *SQLAlchemy Documentation Release 1.4.* SQLAlchemy. Obtenido de https://docs.sqlalchemy.org/en/14/

Benjamín, M. (2014). *Estimadores para modelos con errores en las variables que describen un problema metrológico.* Buenos Aires .

Bustos, C., & Chacón, G. (2007). *El MRP En la gestión de inventarios.* Mérida. Obtenido de https://www.redalyc.org/pdf/4655/465545875010.pdf

Calderón, D. (2015). *Diagnóstico de las variables de incidencia en la fijación de precios de las casas en el sector del Valle de los Chillos, 2010-2014.*

Carro, R., & Gonzáles, D. (2013). *Gestión de stocks* . Universidad Nacional de Mar de Platas .

Carvajal, C., Solís, L., Burgos, I., & Hermida, L. (2017). *La importancia de las PYMES en el Ecuador.* Ecuador: Revista Observatorio de la Economía. Obtenido de http://www.eumed.net/cursecon/ecolat/ec/2017/pymes-ecuador.html

Castillo, J. (2014). *Definición de stock de seguridad y punto de reorden para la compra de equipos en una empresa del servicios del sector telecomunicaciones.* Colombia: Universidad Miliar Nueva Granada.

Castillo, K. (2005). *Propuesta de política de invetarios para productos "A" de la empresa REFA Mexicana S.A. de C.V.* México: Universidad de las Américas Puebla.

Chávez, D. (2021). *Optimización de los recursos de la empresa mediante la aplicación de Modelos de Inventarios.* Tulcán: Universidad Politécnica Estatal del Carchi. Obtenido de http://repositorio.upec.edu.ec/handle/123456789/1719

Córdoba, M. (2014). *Metodología para la toma de desiciones.* Madrid: delta publicaciones.

Cortina, M. L. (2016). *Aplicación y optimización de recursos.* Nuevo Laredo: Universidad Tecnologica de Nuevo Laredo .

Cruz, A. (2017). *UF0476: Gestión de inventarios.* Málaga: IC Editorial.

Díaz, Y., García, B., & Challenge, I. (2014). *El lenguaje de programación Python.* Cuba: Ciencias Holguín.

Guerrero, H. (2022). *Inventarios, manejo y control (3ª ed.)* (Tercera ed.). Bogotá: Ecoe Ediciones.

Heizer, J., & Render, B. (2014). *Principios de administración de operaciones.* Pearson Education.

Hernandez, R., Carlos, C., & Lucio, B. (2014). *Metodologia de la Investigacion.* México: McGraw-Hill /interamericana editores, S.A. De C.V.

Herrera, M. (2017). *Propuesta de un modelo de optimización de recursos para mejorar la eficiencia en el proceso de transformación del plástico.* Bogotá: Universidad Católica de Colombia. Obtenido de https://repository.ucatolica.edu.co/server/api/core/bitstreams/72ba55f5-c27a-419b-9148-19024ee29882/content

Lupo, M., & Rigalli, A. (s.f.). *Mediciones y errores.* Obtenido de https://rephip.unr.edu.ar/bitstreams

Manguay, B., & Vallejo, J. (2022). *Algoritmo de inteligencia artificial en la producción de huevos.* Tulcan: Universidad Politécnica Estatal del Carchi. Obtenido de http://repositorio.upec.edu.ec/handle/123456789/1707

McKinney, W. (2010). *Data Structures for Statistical Computing in Python.* Proceedings of the 9th Python in Science Conference, 51–56. Obtenido de https://doi.org/10.25080/Majora-92bf1922-00a

Meana, P. P. (2017). *Gestion de inventarios UF0476.* Madrid: Cimapress.

Microsoft. (2021). *Documentation for Visual Studio Code*. Obtenido de https://code.visualstudio.com/docs

Puetate, G. (2014). *El Comercio Electrónico y las PYMEs en la ciudad de Tulcán.* Tulcan: Universidad Politécnica Estatal del Carchi. Obtenido de http://repositorio.upec.edu.ec/handle/123456789/61

Redondo, S. (2019). *Why Python Is the Programming Language of the Future.* Towards Data Science. Obtenido de

https://towardsdatascience.com/why-python-is-the-programming-language-of-the-future-3c78942a5dfb

Serrano, J. E. (2019). *Logística de almacenamiento* . Madrid: Paraninfo.

Stackscale. (13 de Septiembre de 2023). *Los lenguajes de programación más populares de 2023* Obtenido de Stackscale: https://www.stackscale.com/es/blog/lenguajes-programacion-mas-populares/

Taha, H. A. (2004). *Investigación de operaciones 7a. edición* . México: PEARSON EDUCACIÓN,.

Tapia, C. (2014). *La cadena de suministros desde la perspectiva de sistema.* España. Obtenido de https://hablemosdelogistics.blogspot.com/2014/03/

Tejero, J. J. (2008). *Almacenes Analisis, diseño y organización* . Madrid: ESIC.

Valera, M. (2023). *Gestión logística y la optimización de recursos de la empresa Grupo ABAN SAC, Lima, 2022.* Lima: Universidad Cesar Vallejo. Obtenido de https://repositorio.ucv.edu.pe/handle/20.500.12692/121625

Vásconez, V. H., Mayorga, M. J., Moreno, M. A., & Arellano, A. V. (2020). *Gestión del sistema de inventarios orientado a pequeñas y medianas empresas, PYMEs, ecuatorianas del sector ferretero:.* Guayaquil: Revista Espacios.

Vásquez, D. (2020). *Gestión de inventarios y su optimización: una revisión de la literatura científica.* Cajamarca: Universidad Privada del Norte. Obtenido de https://hdl.handle.net/11537/25923

Yuni, J., & Urbano, C. (2005). *Mapas y herramientas para conocer la escuela: Investigación etnográfica. Investigación Acción.* Argentina : 3era edición.

VII. ANEXOS

Anexo 1. Acta de la sustentación de Predefensa del TIC

UNIVERSIDAD POLITÉCNICA ESTATAL DEL CARCHI

FACULTAD DE COMERCIO INTERNACIONAL, INTEGRACIÓN, ADMINISTRACIÓN Y ECONOMÍA EMPRESARIAL

CARRERA DE LOGÍSTICA Y TRANSPORTE

ACTA

DE LA SUSTENTACIÓN ORAL DE LA PREDEFENSA DEL TRABAJO DE INTEGRACIÓN CURRICULAR

ESTUDIANTE:	RUANO PISCAL LUIS STIVEN	**CÉDULA DE IDENTIDAD:**	0450071592
PERIODO ACADÉMICO:	2025A		
PRESIDENTE TRIBUNAL	MSc. Beltrán del Hierro Daniel Mauricio	**DOCENTE TUTOR:**	MSc. Mafla Bolaños Iván Gabriel
DOCENTE:	MSc. Alpala Alpala Luis Omar		
TEMA DEL TIC:	"Diseño de un programa para la gestión de inventarios basado en árbol de decisiones que permita la optimización de recursos en las empresas"		

No.	CATEGORÍA	Evaluación cuantitativa	OBSERVACIONES Y RECOMENDACIONES
1	PROBLEMA - OBJETIVOS	10,00	
2	FUNDAMENTACIÓN TEÓRICA	10,00	
3	METODOLOGÍA	10,00	
4	RESULTADOS	8,67	Complemetar el análisis de optimización de recursos
5	DISCUSIÓN	9,00	Complementar con el resultado sugerido
6	CONCLUSIONES Y RECOMENDACIONES	9,00	Complementar con el resultado sugerido
7	DEFENSA, ARGUMENTACIÓN Y VOCABULARIO PROFESIONAL	8,00	Controlar vocalización, la presentación debe ser pausada para que se comprenda lo expuesto
8	FORMATO, ORGANIZACIÓN Y CALIDAD DE LA INFORMACIÓN	9,00	Revisar redacción de documento y formato de paginas preliminares

Obteniendo una nota de: 9,17 Por lo tanto, **APRUEBA** ; debiendo el o los investigadores acatar el siguiente artículo:

Art. 36.- De los estudiantes que aprueban el informe final del TIC con observaciones.- Los estudiantes tendrán el plazo de 10 días para proceder a corregir su informe final del TIC de conformidad a las observaciones y recomendaciones realizadas por los miembros del Tribunal de sustentación de la pre-defensa.

Para constancia del presente, firman en la ciudad de Tulcán el jueves, 27 de febrero de 2025

MSc. Beltrán del Hierro Daniel Mauricio
PRESIDENTE TRIBUNAL

MSc. Mafla Bolaños Iván Gabriel
DOCENTE TUTOR

MSc. Alpala Alpala Luis Omar
DOCENTE

UNIVERSIDAD POLITÉCNICA ESTATAL DEL CARCHI

FACULTAD DE COMERCIO INTERNACIONAL, INTEGRACIÓN, ADMINISTRACIÓN Y ECONOMÍA EMPRESARIAL

CARRERA DE LOGÍSTICA Y TRANSPORTE

ACTA

DE LA SUSTENTACIÓN ORAL DE LA PREDEFENSA DEL TRABAJO DE INTEGRACIÓN CURRICULAR

ESTUDIANTE:	TAPIA COLLAGUAZO HENRY STALIN	**CÉDULA DE IDENTIDAD:**	1724754724
PERIODO ACADÉMICO:	2025A		
PRESIDENTE TRIBUNAL	MSc. Beltrán del Hierro Daniel Mauricio	**DOCENTE TUTOR:**	MSc. Mafla Bolaños Iván Gabriel
DOCENTE:	MSc. Alpala Alpala Luis Omar		
TEMA DEL TIC:	"Diseño de un programa para la gestión de inventarios basado en árbol de decisiones que permita la optimización de recursos en las empresas"		

No.	CATEGORÍA	Evaluación cuantitativa	OBSERVACIONES Y RECOMENDACIONES
1	PROBLEMA - OBJETIVOS	10,00	
2	FUNDAMENTACIÓN TEÓRICA	10,00	
3	METODOLOGÍA	10,00	
4	RESULTADOS	8,67	Complemetar el análisis de optimización de recursos
5	DISCUSIÓN	9,00	Complementar con el resultado sugerido
6	CONCLUSIONES Y RECOMENDACIONES	9,00	Complementar con el resultado sugerido
7	DEFENSA, ARGUMENTACIÓN Y VOCABULARIO PROFESIONAL	9,00	Controlar vocalización, la presentación debe ser pausada para que se comprenda lo expuesto
8	FORMATO, ORGANIZACIÓN Y CALIDAD DE LA INFORMACIÓN	9,00	Revisar redacción de documento y formato de paginas preliminares

Obteniendo una nota de: 9,27 Por lo tanto, **APRUEBA** ; debiendo el o los investigadores acatar el siguiente artículo:

Art. 36.- De los estudiantes que aprueban el informe final del TIC con observaciones.- Los estudiantes tendrán el plazo de 10 días para proceder a corregir su informe final del TIC de conformidad a las observaciones y recomendaciones realizadas por los miembros del Tribunal de sustentación de la pre-defensa.

Para constancia del presente, firman en la ciudad de Tulcán el jueves, 27 de febrero de 2025

MSc. Beltrán del Hierro Daniel Mauricio
PRESIDENTE TRIBUNAL

MSc. Mafla Bolaños Iván Gabriel
DOCENTE TUTOR

MSc. Alpala Alpala Luis Omar
DOCENTE

Anexo 2. Certificado del *abstract* por parte de idiomas

UNIVERSIDAD POLITÉCNICA ESTATAL DEL CARCHI FOREIGN AND NATIVE LANGUAGES CENTER

ABSTRACT- EVALUATION SHEET

NAME: Ruano Piscal Luis Stiven y Tapia Collaguazo Henry Stalin.
DATE: Jueves, 13 de marzo de 2025
Topic: Diseño de un programa para la gestión de inventarios basado en árbol de decisiones que permita la optimización de recursos en las empresas
MARKS AWARDED **QUANTITATIVE AND QUALITATIVE**

VOCABULARY AND WORD USE	Use new learnt vocabulary and precise words related to the topic	Use a little new vocabulary and some appropriate words related to the topic	Use basic vocabulary and simplistic words related to the topic	Limited vocabulary and inadequate words related to the topic
	EXCELLENT: 2 ☐	GOOD: 1,5 ■	AVERAGE: 1 ☐	LIMITED: 0,5 ☐
WRITING COHESION	Clear and logical progression of ideas and supporting paragraphs.	Adequate progression of ideas and supporting paragraphs.	Some progression of ideas and supporting paragraphs.	Inadequate ideas and supporting paragraphs.
	EXCELLENT: 2 ■	GOOD: 1,5 ☐	AVERAGE: 1 ☐	LIMITED: 0,5 ☐
ARGUMENT	The message has been communicated very well and identify the type of text	The message has been communicated appropriately and identify the type of text	Some of the message has been communicated and the type of text is little confusing	The message hasn't been communicated and the type of text is inadequate
	EXCELLENT: 2 ■	GOOD: 1,5 ☐	AVERAGE: 1 ☐	LIMITED: 0,5 ☐
CREATIVITY	Outstanding flow of ideas and events	Good flow of ideas and events ■	Average flow of ideas and events	Poor flow of ideas and events
	EXCELLENT: 2 ☐	GOOD: 1,5	AVERAGE: 1 ☐	LIMITED: 0,5 ☐
SCIENTIFIC SUSTAINABILITY	Reasonable, specific and supportable opinion or thesis statement	Minor errors when supporting the thesis statement	Some errors when supporting the thesis statement	Lots of errors when supporting the thesis statement
	EXCELLENT: 2 ■	GOOD: 1,5 ☐	AVERAGE: 1 ☐	LIMITED: 0,5 ☐

TOTAL/AVERAGE

9 - 10: EXCELLENT
7 - 8,9: GOOD
5 - 6,9: AVERAGE
0 - 4,9: LIMITED

TOTAL 9

UNIVERSIDAD POLITÉCNICA ESTATAL DEL CARCHI- FOREIGN AND NATIVE LANGUAGES CENTER

Informe sobre el Abstract de Artículo Científico o Investigación.

Autor: Ruano Piscal Luis Stiven y Tapia Collaguazo Henry Stalin.
Fecha de recepción del abstract: Miércoles, 12 de marzo de 2025
Fecha de entrega del informe: Jueves, 13 de marzo de 2025

El presente informe validará la traducción del idioma español al inglés si alcanza un porcentaje de: 9 – 10 Excelente.

Si la traducción no está dentro de los parámetros de 9 – 10, el autor deberá realizar las observaciones presentadas en el ABSTRACT, para su posterior presentación y aprobación.

Observaciones:

Después de realizar la revisión del presente abstract, éste presenta una apropiada traducción sobre el tema planteado en el idioma Inglés. Según la rúbrica de evaluación de la traducción en Inglés, ésta alcanza un valor de 9; por lo cual se valida dicho trabajo.

Atentamente

MARTHA ARACELLY VIVEROS ALMEIDA

MA. Martha Viveros
Docente responsable del
CIDEN

Anexo 3. Entrevista - Guía de preguntas

Guía de preguntas

Empresa	
Persona entrevistada	
Fecha	

Compras

1. ¿Cómo se gestionan los gastos adicionales asociados con las compras, como impuestos y tarifas de envío?
2. ¿Cómo se realiza el proceso de compras, y que parámetros se tienen en cuenta, como proveedores, tiempos de entrega, etc?

Almacenamiento

1. ¿Cuál es el procedimiento para el almacenamiento de productos?
2. ¿Cómo se organiza y gestiona el movimiento de productos en el almacén? ¿Existen sectores designados para cada tipo de producto?
3. ¿Cómo se maneja la clasificación y categorización de los productos en inventario?

Ventas

1. ¿Cómo se comporta la demanda de sus productos?
2. ¿Cómo se realiza el proceso de ventas, y que parámetros se tienen en cuenta, como clientes preferenciales, ventas online, pedidos, tiempos de entrega, etc.?
3. ¿La empresa realiza ventas mediante pedidos anticipados?

Uso Eficiente de Inventarios.

1. ¿Con qué frecuencia se realiza el conteo físico del inventario?
2. ¿Se calculan indicadores como rotación de inventario periódicamente?
3. ¿Con qué frecuencia se presentan situaciones de desabasto o faltantes?
4. ¿Se mantienen niveles excesivos de ciertos productos frecuentemente?

5. ¿Se considera que se incurre en costos innecesarios por el manejo del inventario?
6. ¿Cómo se registran actualmente las entradas y salidas de inventario?
7. ¿Existe alguna documentación formal que respalde los registros de inventario?
8. ¿Qué productos tiene actualmente disponibles en su inventario?
9. ¿Se utiliza el modelo de Cantidad Económica de Pedido y si es así como la determinan para productos específicos?
10. ¿Cómo se determina el Punto de Reorden para productos específicos?
11. ¿Se utiliza algún sistema tecnológico para llevar el control del inventario?
12. ¿Se utiliza algún sistema tecnológico para los procesos de compras, almacenamiento y ventas?
13. ¿Se utilizan herramientas o sistemas para agilizar el proceso de seguimiento de clientes potenciales y conversiones?
14. ¿Qué herramientas se utilizan para llevar un seguimiento de los niveles de inventario?
15. ¿Cómo se gestionan las discrepancias o faltantes identificados durante el inventario?

Mejora en la Planificación y Programación.

1. ¿Todos los pedidos realizados son atendidos de manera satisfactoria por la empresa?
2. ¿Cómo se identifican y manejan los posibles retrasos en la planificación?
3. ¿Existen sistemas o herramientas específicas para el monitoreo continuo de los plazos y pedidos?
4. ¿Cómo se define y documenta actualmente el proceso de planificación y programación para los plazos y pedidos?

Anexo 4. Ficha técnica para modelos de inventarios

Ficha de análisis 1	
Modelo	**Modelos estáticos de cantidad económica de pedido (CEP, o *EOQ*):** Modelo clásico de cantidad económica de pedido.
Definición	El modelo de inventario más simple supone una demanda constante y una reposición instantánea, es decir, los productos están disponibles de inmediato una vez realizado el pedido.
Tipo de demanda	Determinística
Parámetros	• Cantidad pedida • Tasa de demanda • Costo de preparación de pedido • Costo unitario de almacenamiento por unidad de tiempo • Tiempo de entrega
Tipo de sistema de revisión	Revisión periódica
Ventajas	Sencillo de entender y aplicar
Desventajas	Supuestos simplificados, no considera incertidumbre en la demanda ni en el tiempo de entrega

Ficha de análisis 2	
Modelo	**Modelos estáticos de cantidad económica de pedido (CEP, o *EOQ*):** Cantidad económica de pedido con discontinuidades de precio
Definición	Este modelo es una variante del Modelo clásico de cantidad económica de pedido, pero con la diferencia de que permite obtener descuentos en el precio unitario de compra si el tamaño del pedido supera un límite específico q.
Tipo de demanda	Determinística
Parámetros	• Cantidad pedida • Tasa de demanda • Costo de preparación de pedido • Costo unitario de almacenamiento por unidad de tiempo • Tiempo de entrega • Tamaño del pedido con descuento • Costo normal del pedido • Costo con descuento del pedido
Tipo de sistema de revisión	Revisión periódica
Ventajas	Permite optimizar la cantidad de pedido considerando descuentos por volumen
Desventajas	Requiere información precisa sobre la estructura de descuentos por cantidad

Ficha de análisis 3	
Modelo	**Modelos estáticos de cantidad económica de pedido (CEP, o *EOQ*):**

Definición	Cantidad económica de pedido de varios artículos con limitación de almacén. Este modelo e utiliza en situaciones con varios artículos (n > 1) cuyo inventario varía según un patrón específico, sin permitir faltantes. La particularidad es que los artículos deben compartir un espacio de almacenamiento limitado, compitiendo por el mismo.
Tipo de demanda	Determinística
Parámetros	• Costo unitario de almacenamiento por unidad de tiempo • Área máxima disponible de almacenamiento para los n artículos • Costo de almacenamiento por unidad de inventario y por unidad de tiempo
Tipo de sistema de revisión	Revisión periódica
Ventajas	Permite optimizar la gestión de inventarios para múltiples artículos con limitaciones de espacio
Desventajas	Es un modelo complejo que requiere resolver un sistema de ecuaciones

Ficha de análisis 4	
Modelo	**Modelos dinámicos de cantidad económica de pedido:** Modelo sin costo de preparación
Definición	Este modelo considera un horizonte de planificación dividido en N períodos de igual duración y capacidad de producción limitada en cada uno, con varios niveles de producción (como tiempo normal y tiempo extra). Si en un periodo se produce más de lo demandado, el exceso se almacena para periodos posteriores, incurriendo en un costo de almacenamiento.
Tipo de demanda	Determinístico
Parámetros	• Oferta acumulada • Demanda acumulada
Tipo de sistema de revisión	Revisión periódica
Ventajas	Sencillo de aplicar en situaciones donde no hay costos de preparación
Desventajas	No considera los costos fijos asociados a la realización de pedidos

Ficha de análisis 5	
Modelo	**Modelos dinámicos de cantidad económica de pedido:** Modelo con preparación
Definición	Este modelo considera un horizonte de planificación dividido en N períodos de igual duración y capacidad de

	producción limitada en cada uno, con varios niveles de producción (como tiempo normal y tiempo extra). Si en un periodo se produce más de lo demandado, el exceso se almacena para periodos posteriores, incurriendo en un costo de almacenamiento.
Tipo de demanda	Determinístico
Parámetros	• Cantidad pedida • Tasa de demanda • Costo de preparación de pedido • Costo unitario de almacenamiento por unidad de tiempo • Demanda para periodo i • Inventario al inicio del periodo i
Tipo de sistema de revisión	Revisión periódica
Ventajas	Más realista que el modelo sin costo de preparación
Desventajas	Requiere información precisa sobre el costo de preparación por pedido

Ficha de análisis 6	
Modelo	**Modelos de revisión continua:** Modelo "Probabilizado" de cantidad económica de pedido
Definición	Una extensión del modelo *EOQ* clásico que considera incertidumbre en la demanda. Algunos profesionales han adaptado el modelo determinístico de cantidad económica de pedido para reflejar la naturaleza probabilista de la demanda, añadiendo una reserva constante al nivel de inventario durante todo el horizonte de planeación. Esta reserva se dimensiona para que la probabilidad de agotarse durante el tiempo de entrega no supere un valor especificado.
Tipo de demanda	Probabilístico
Parámetros	• Tasa de demanda • Tiempo de entrega • Variable aleatoria que representa la demanda durante el tiempo de entrega • Demanda promedio durante el tiempo de entrega • Desviación estándar de la demanda durante el tiempo de entrega • Tamaño de la existencia de reserva • Probabilidad máxima admisible de que se agote la existencia • Faltante por unidad
Tipo de sistema de revisión	Revisión periódica
Ventajas	Permite considerar la incertidumbre en la demanda
Desventajas	Requiere información sobre la distribución de probabilidad de la demanda

Ficha de análisis 7	
Modelo	**Modelos de revisión continua:** Modelo probabilístico de cantidad económica de pedido
Definición	Es parecido al modelo *EOQ*, pero acepta la ocurrencia de faltantes de inventario. Emplea una política de reorden en la que se realiza un pedido cuando el inventario desciende a un nivel R. Los valores óptimos de la cantidad a pedir (y) y el nivel de reorden (R) se determinan minimizando el costo esperado por unidad de tiempo, que incluye los costos de preparación, almacenamiento y faltantes.
Tipo de demanda	Probabilístico
Parámetros	• Tasa de demanda • Tiempo de entrega • Costo de preparación de pedido • Costo unitario de almacenamiento por unidad de tiempo • Demanda esperada por unidad de tiempo • Costo de almacenamiento por unidad de inventario y por unidad de tiempo • Costo de faltante por unidad de inventario • Función de distribución de probabilidades de la demanda x durante el tiempo de entrega.
Tipo de sistema de revisión	Revisión periódica
Ventajas	Más flexible que el modelo *EOQ*, permite establecer un nivel de servicio objetivo
Desventajas	Requiere información sobre la distribución de probabilidad de la demanda y el nivel de servicio deseado

Ficha de análisis 8	
Modelo	**Modelos de un periodo:** Modelo sin preparación
Definición	Es un caso particular del modelo *EOQ* en el que no se generan costos fijos por realizar pedidos. En este modelo, la demanda se presenta de manera instantánea al inicio del período, justo después de recibir el pedido. Además, no se incurre en costos de preparación.
Tipo de demanda	Probabilístico
Parámetros	• Cantidad pedida • Costo de preparación de pedido • Costo de compra por unidad • Costo de almacenamiento por unidad conservada durante el periodo • Penalización por unidad faltante durante el periodo • Cantidad a la mano, antes de hacer un pedido

Ficha de análisis 8	
	• Variable aleatoria que representa la demanda durante el periodo • Distribución de la función de probabilidad de la demanda durante el periodo
Tipo de sistema de revisión	Revisión periódica
Ventajas	Sencillo de aplicar en situaciones donde no hay costos de preparación
Desventajas	No considera los costos fijos asociados a la realización de pedidos

Ficha de análisis 9	
Modelo	**Modelos de un periodo:** Modelo con preparación (política s-S)
Definición	Este modelo de inventario tiene en cuenta un costo fijo asociado a la realización de pedidos y un nivel de inventario de seguridad. Es adecuado para escenarios en los que la demanda varía y hay riesgo de desabastecimientos. A diferencia del modelo presentado en la sección "Modelo sin preparación", este incluye un costo de preparación K. Usando la misma notación, se determina el costo total esperado por cada período.
Tipo de demanda	Probabilístico
Parámetros	• Cantidad pedida • Costo de preparación de pedido • Costo de compra por unidad • Costo de almacenamiento por unidad conservada durante el periodo • Penalización por unidad faltante durante el periodo • Cantidad a la mano, antes de hacer un pedido • Variable aleatoria que representa la demanda durante el periodo
Tipo de sistema de revisión	Revisión periódica
Ventajas	Permite reducir los costos de faltantes de inventario, mantener un nivel de servicio objetivo.
Desventajas	Requiere más información para implementarse, puede ser menos eficiente en situaciones donde la demanda es muy variable

Ficha de análisis 10	
Modelo	Modelos de varios periodos
Definición	Este modelo contempla la planificación de inventarios a lo largo de un horizonte temporal discreto, dividiéndolo en

Ficha de análisis 10	
	períodos más cortos. En el caso de varios períodos, se asume que no existe costo de preparación y se permite acumular la demanda sin que haya demoras en la entrega. La demanda en cada período se modela mediante una función de distribución de probabilidades estacionaria.
Tipo de demanda	Probabilístico
Parámetros	• Cantidad pedida • Costo de compra por unidad • Costo de almacenamiento por unidad conservada durante el periodo • Penalización por unidad faltante durante el periodo • Cantidad a la mano, antes de hacer un pedido • Variable aleatoria que representa la demanda durante el periodo • Ingreso por unida • Precio de venta por unidad • Factor de descuento • Variable aleatoria que representa la demanda durante el periodo • Costo de penalización por unidad y unidad de tiempo • Distribución de la función de probabilidad de la demanda durante el periodo
Tipo de sistema de revisión	Revisión periódica o continua
Ventajas	Permiten optimizar la gestión de inventarios a lo largo de un horizonte de tiempo más largo
Desventajas	Son modelos más complejos que requieren más información y mayor capacidad computacional

Anexo 5. Códigos por cada modelo de inventario

Modelos determinísticos

Modelo clásico de cantidad económica de pedido

```
import math

# Función para calcular la cantidad económica de pedido (EOQ)
def calcular_eoq(K, D, h):
    """
    Calcula la cantidad económica de pedido (EOQ).

    Parámetros:
```

```
    K (float): Costo de pedido.
    D (float): Demanda mensual.
    h (float): Costo de mantenimiento por unidad por mes.

    Retorna:
    float: Cantidad económica de pedido (EOQ).
    """
    return math.sqrt((2 * K * D) / h)

# Función para calcular el costo total mensual
def calcular_costo_total_mensual(K, D, h, Q):
    """
    Calcula el costo total mensual.

    Parámetros:
    K (float): Costo de pedido.
    D (float): Demanda mensual.
    h (float): Costo de mantenimiento por unidad por mes.
    Q (float): Cantidad económica de pedido (EOQ).

    Retorna:
    float: Costo total mensual.
    """
    costo_total_mensual = K /((math.sqrt((2 * K * D) / h))/D) + h * (math.sqrt((2 * K * D) / h))/2

    return costo_total_mensual

# Función para calcular el punto de reorden
def calcular_punto_reorden(D, L):
    """
    Calcula el punto de reorden.

    Parámetros:
    D (float): Demanda mensual.
    L (float): Tiempo de entrega en meses.

    Retorna:
    float: Punto de reorden.
    """
    to = (math.sqrt((2 * K * D) / h))/D
    n = math.floor(L / to)
    Le = L - n*to
    LeD = Le * D
    return LeD
```

```
# Parámetros del modelo
D = 832  # Demanda mensual
K = 69.82    # Costo de pedido
h = 2.96   # Costo de mantenimiento por unidad por mes
L = 5    # Tiempo de entrega en meses

# Cálculo del EOQ
eoq = calcular_eoq(K, D, h)
print(f"Cantidad económica de pedido (EOQ): {eoq:.2f} unidades")

# Cálculo del costo total mensual
costo_total_mensual = calcular_costo_total_mensual(K, D, h, eoq)
print(f"Costo total mensual: ${costo_total_mensual:.2f}")

# Cálculo del punto de reorden
punto_reorden = calcular_punto_reorden(D, L)
print(f"Punto de reorden: {punto_reorden:.2f} unidades")
```

Cantidad económica de pedido con discontinuidades de precio

```
import math

D = 187.5  #Tasa de demanda
k = 20     # Costo preparacion de pedido
c1 = 3     # Costo de compra normal
c2 = 2.50   # Costo de compra con descuento
h =0.02    # Costo de almacenamiento
q = 1000  # Tamaño minimo del pedido con descuento
L =2       # Tiempo de entrega

y_punto = ((2*k*D)/h)**(0.5)

to = (math.sqrt((2 * k * D) / h))/D
n = math.floor(L / to)
Le = L - n*to
y_r = Le * D

TCU= D*c1 + k*D/y_punto + (h*y_punto/2)

def solve_quadratic(a, b, c, threshold):
    # Calcular el discriminante
    discriminant = b**2 - 4*c

    # Verificar si el discriminante es negativo, sin soluciones reales
```

```
    if discriminant < 0:
        return "No hay soluciones reales"

    # Calcular las dos posibles soluciones
    x1 = (-b + math.sqrt(discriminant)) / (2)
    x2 = (-b - math.sqrt(discriminant)) / (2)

    # Devolver la solución que sea mayor que el umbral
    if x1 > y_punto and x2 > y_punto:
        return max(x1, x2)
    elif x1 > y_punto:
        return x1
    elif x2 > y_punto:
        return x2
    else:
        return "No hay solución mayor que el umbral"

# Ejemplo de uso con los valores proporcionados
f = 2  # Coeficiente de x^2
b = ((L*(c2*D-TCU))/h)
c = (L*k*D/h)

resultado = solve_quadratic(f, b, c, y_punto)

if q > y_punto and q < resultado:
  y_opt= q
else:
  y_opt=y_punto

print(f"Cantidad economica de pedido: {y_opt} ,  con un punto de reorden de {y_r}")
```

Cantidad económica de pedido de varios artículos con limitación de almacén

```
import numpy as np
def inv_restriccion_area(k,D,h,a,A,error = 0.05):
 y_sub = []
 for i in range(0,n):
   yi= ((2*k[i]*D[i])/(h[i]))**0.5
   y_sub.append(yi)
 area_ocupada = np.dot(a,y_sub)
 if area_ocupada <= A:
   print("Se ha encontrado la sol óptima con y_punto")
   return y_sub,area_ocupada
 else:
   print("Calcular con Lagrange")
```

```
    diferencia = 10
    lamb = 0
    area_ocu_op = 0

    while diferencia > error:
      y_opt = []

      for j in range(0,n):
        y_optimo =((2*k[j]*D[j])/(h[j]-2*lamb*a[j]))**0.5
        y_opt.append(y_optimo)
      area_ocu_op = np.dot(a,y_opt)
      if area_ocu_op <= A:
        diferencia = np.abs(A - area_ocu_op)
        lamb -= 0.001
      lamb -= 0.001
    print("La solución óptima es:")
    return y_opt,area_ocu_op

## dATOS
k = [10,5,15,6]
D = [2,4,4,5]
h = [0.3,0.1,0.2,0.3]
a = [1,1,1,2]
n = len(k)
A = 30

sol_opt,area_ocupada = inv_restriccion_area(k,D,h,a,A,error = 0.1)
for yi in range(0,n):
  print(f"El pedido óptimo del artículo {yi + 1} es: {round(sol_opt[yi],2)} unidades")
print(f"El área total ocupada es {round(area_ocupada,3)} unidades cuadradas y es menor o igual a {A}")

np.dot(a,k)
```

Modelo sin costo de preparación

```
import numpy as np

# Demand for each period
demand = np.array([100, 190, 210, 160])

# Normal production capacities per period
normal_capacity = np.array([90, 100, 120, 110])
```

```
# Overtime production capacities per period
overtime_capacity = np.array([50, 60, 80, 70])

# Costs per unit
normal_cost = 6.0
overtime_cost = 9.0
storage_cost = 0.10  # Monthly storage cost per unit

# Number of periods
num_periods = len(demand)

# Initialize cost and production matrices
total_cost = 0
production_schedule = np.zeros((num_periods, 2), dtype=int)  # [Normal, Overtime] production
stored_units = np.zeros(num_periods, dtype=int)

# Production planning
for i in range(num_periods):
    if i == 0:
        # Produce in normal time
        production_schedule[i, 0] = normal_capacity[i]
        remaining_demand = demand[i] - normal_capacity[i]
        if remaining_demand > 0:
            # Produce remaining demand in overtime
            production_schedule[i, 1] = remaining_demand
        stored_units[i] = normal_capacity[i] + production_schedule[i, 1] - demand[i]
    else:
        # Produce in normal time
        if stored_units[i-1] < demand[i]:
            required_production = demand[i] - stored_units[i-1]
            production_schedule[i, 0] = min(required_production, normal_capacity[i])
            remaining_demand = required_production - production_schedule[i, 0]
            if remaining_demand > 0:
                # Produce remaining demand in overtime
                production_schedule[i, 1] = remaining_demand
        stored_units[i] = production_schedule[i, 0] + production_schedule[i, 1] + stored_units[i-1] - demand[i]

    # Compute total cost
    normal_production_cost = production_schedule[i, 0] * normal_cost
    overtime_production_cost = production_schedule[i, 1] * overtime_cost
    storage_cost_total = stored_units[i] * storage_cost
    total_cost += normal_production_cost + overtime_production_cost + storage_cost_total

# Output results
```

```
print("Programa de producción óptimo:")
for i in range(num_periods):
    print(f"Periodo {i+1}: Producir {production_schedule[i, 0]} unidades en tiempo normal y {production_schedule[i, 1]}
unidades en tiempo extra")
print(f"Costo total de producción y almacenamiento: ${total_cost:.2f}")
```

Modelo con preparación

```
import numpy as np

## Datos del problema
D = [3,2,4]
K = [3,7,6]
h = [1,3,2]
x_inicial = 1
c_pro1 = 10
c_pro2 = 20
unidades_min = 3


max = sum(D)

## número de etapas n
n = len(D)

## Función de costo marginal de producción
def c_i(p_i):
  if p_i <= unidades_min:
    return c_pro1 * p_i
  else:
    return  (c_pro1 + c_pro2) + c_pro2 * (p_i - 3)

## Función de costo de producción
def C_i(p_i, i):
  if p_i == 0:
    return 0
  else:
    return K[i] + c_i(p_i)

## Inicializar las tablas de costos y las soluciones óptimas
f = np.full((n + 1,max),np.inf)  ## costos mínimos
x_opt = np.zeros((n + 1,max), dtype = int)  ## Inventarios óptimos

## Condición inicial  n + 1
f[n, :] = 0  # Costo mínimo en el último periodo es 0
```

```
## Programación dinámica en reversa
for i in range(n - 1, -1, -1):
  for x in range(max):
    for p in range(max):
      x_next = x + p - D[i]
      if 0 <= x_next < max:
        costo = C_i(p,i) + h[i] * x + f[i + 1, x_next]
        if costo < f[i,x]:
          f[i , x] = costo
          x_opt[i, x] = p

## Reconstruir la solución óptima
p_opt = []
x = x_inicial
for i in range(n):
  p_opt.append(x_opt[i,x])
  x = x + x_opt[i , x] - D[i]

## Resultados
for i in range(1, n + 1):
  print(f"En el periodo {i} pedir {p_opt[i - 1]} unidades")
print(f"El costo mínimo de inventario MRP es {f[0,x_inicial]}")
```

MODELOS DE INVENTARIOS PROBABILISTICOS

Modelo “probabilizado” de cantidad económica de pedido

```
import scipy.stats as stats
import math

def calcular_eoq_probabilizado(D, sigma, L, alpha):
    # Media de la demanda durante el tiempo de entrega
    mu_L = D * L

    # Desviación estándar de la demanda durante el tiempo de entrega
    sigma_L = sigma * math.sqrt(L)

    # Calcular el valor Z para el nivel de servicio deseado
    Z = stats.norm.ppf(1 - alpha)

    # Calcular el stock de seguridad
    B = Z * sigma_L
```

```
    # Punto de reorden
    R = mu_L + B

    return R, B

# Ejemplo de uso
D = 176  # Demanda diaria promedio (unidades por día)
sigma = 29  # Desviación estándar de la demanda diaria (unidades por día)
L =  1   # Tiempo de entrega (días)
alpha = 0.02  # Probabilidad máxima admisible de agotamiento de existencias

R, B = calcular_eoq_probabilizado(D, sigma, L, alpha)
print(f"Punto de reorden (R): {R:.2f} unidades")
print(f"Stock de seguridad (B): {B:.2f} unidades")
```

Modelo probabilístico de cantidad económica de pedido

```
import math as m
import sympy as sp

# Parámetros
K = 100  # Costo de preparación de pedido
h = 2    # Costo de almacenamiento
p = 10   # Costo de faltante por unidad de inventario
D = 1000 # Demanda
fx = 1 / 100
denominador_fx = 100
Ex = 50

# Calcular y1 y y2
y1 = m.sqrt(2 * D * (K + p * Ex) / h)
y2 = (p * D) / h

# Condición para continuar con los cálculos
if y2 >= y1:
    # Definir la variable x y el parámetro R
    x, R, v = sp.symbols('x R v')

    # Definir los límites de integración
    limite_inferior = R

    # Definir la integral simbólica S
    S = sp.integrate((x - R) * fx, (x, limite_inferior, denominador_fx))

    # Simplificar el resultado para obtener una ecuación cuadrática en R
```

```
S_simplificada = sp.simplify(S)

# Calcular yi
yi = sp.sqrt((2 * D * (K + p * v)) / h)  # Aquí 'v' se mantiene como variable

# Calcular la nueva yi1
yi1 = sp.symbols('yi1')  # Definir yi1 como símbolo

# Calcular la integral
z = denominador_fx
integral_result = sp.integrate(fx, (x, R, z))

# Resolver la ecuación
ecuacion = sp.Eq(integral_result, (h * yi1) / (p * D))

# Despejar R
R_resultado = sp.solve(ecuacion, R)[0]

# Mostrar el resultado

yn1 = m.sqrt(2*K*D/h)

# Reemplazar yi1 con yn1 en la ecuación de R
nueva_ecuacion = ecuacion.subs(yi1, yn1)

# Resolver para obtener el nuevo R
nuevo_R = sp.solve(nueva_ecuacion, R)[0]


#ITERACION
 # Reemplazar R en S_simplificada con nuevo_R
S_final = S_simplificada.subs(R, nuevo_R)

y_it1= yi.subs(v,S_final)

nueva_ecuacion2 = ecuacion.subs(yi1,y_it1)
nuevo_R2 = sp.solve(nueva_ecuacion2, R)[0]

S_final2 = S_simplificada.subs(R, nuevo_R2)

y_it2= yi.subs(v,S_final2)

nueva_ecuacion3 = ecuacion.subs(yi1,y_it2)
nuevo_R3 = sp.solve(nueva_ecuacion3, R)[0]

S_final3 = S_simplificada.subs(R, nuevo_R3)
```

```
    y_it3= yi.subs(v,S_final3)

    nueva_ecuacion4 = ecuacion.subs(yi1,y_it3)
    nuevo_R4 = sp.solve(nueva_ecuacion4, R)[0]

    S_final4 = S_simplificada.subs(R, nuevo_R4)

    y_it4= yi.subs(v,S_final4)

    nueva_ecuacion5 = ecuacion.subs(yi1,y_it4)
    nuevo_R5 = sp.solve(nueva_ecuacion5, R)[0]

    S_final5 = S_simplificada.subs(R, nuevo_R5)

    y_it5= yi.subs(v,S_final5)
    print(f"yi_1:   {y_it5}")

    nueva_ecuacion6 = ecuacion.subs(yi1,y_it5)
    nuevo_R6 = sp.solve(nueva_ecuacion6, R)[0]
    print(f"Ri_1 =  {nuevo_R6}")

    S_final6 = S_simplificada.subs(R, nuevo_R6)

    y_it6= yi.subs(v,S_final6)
    print(f"yi_2:   {y_it6}")

    nueva_ecuacion7 = ecuacion.subs(yi1,y_it6)
    nuevo_R7 = sp.solve(nueva_ecuacion7, R)[0]
    print(f"Ri_2 =  {nuevo_R7}")

    #Resultado
    print("")
    print("POLITICA OPTIMA DE PEDIDO")
    print(f"Se debe pedir aproximadamente {y_it6:2f} unidades, siempre que el nivel de
existencias baje a {nuevo_R7:2f} ")



else:
    print("No cumplió la restricción para seguir con el cálculo.")
```

Modelo sin preparación

```
# MODELO PROBABILISTICO SIN COSTO DE PREPARACION ( DISTRIBUCION DISCRETA)


import numpy as np
```

```
c = 30  # COSTO DE COMPRA
h = 25   # COSTO DE ALMACENAMIENTO
p = 45   # COSTO DE PENALIZACION

if p < c:
    print("No se puede continuar el cálculo porque el costo de compra del artículo es mayor que la penalización por no suministrarlo.")
else:
    # RELACION CRITICA
    RC = (p - c) / (p + h)

# Función para calcular la distribución acumulada
def calcular_distribucion_acumulada(valores, probabilidades, RC):
    # Asegurarse de que las probabilidades sumen 1
    if not np.isclose(sum(probabilidades), 1):
        raise ValueError("Las probabilidades deben sumar 1.")

    distribucion_acumulada = np.cumsum(probabilidades)


    # Determinar el valor más alto en el rango donde se ubica RC
    for i, acumulada in enumerate(distribucion_acumulada):
        if RC <= acumulada:
            # Retornar el valor más alto correspondiente
            return valores[i]

    # Si RC es mayor que la acumulada total, retornar None
    return None

# Introducir valores y probabilidades
Distrib_discreta = [200, 220, 300, 320, 340]  # Valores de la variable aleatoria
FD = [0.1, 0.2, 0.4, 0.2, 0.1]  # Probabilidades asociadas a cada valor

# Calcular y mostrar la distribución acumulada
valor_mas_alto = calcular_distribucion_acumulada(Distrib_discreta, FD, RC)

# Mostrar el resultado
if valor_mas_alto is not None:
    print(f"De acuerdo a la funcion de distribucion de probabilidades discreta la cantidad de pedido es: {valor_mas_alto}")
else:
    print(f"RC = {RC} es mayor que la acumulada total.")




# MODELO PROBABILISTICO SIN COSTO DE PREPARACION (DISTRIBUCION CONTINUA)
```

```
import numpy as np
import scipy.stats as stats

# DISTRIBUCION CONTINUA

c = 30  # COSTO DE COMPRA
h = 25   # COSTO DE ALMACENAMIENTO
p = 45   # COSTO DE PENALIZACION
D = 300  # DEMANDA PROMEDIO
sigma = 20  # DESVIACION ESTANDAR

# VERIFICAR CONDICIÓN
if p < c:
    print("No se puede continuar el cálculo porque el costo de compra del artículo es mayor que la penalización por no suministrarlo.")
else:
    # RELACION CRITICA
    RC = (p - c) / (p + h)

    # CALCULO TABLA DE DISTRIBUCION NORMAL
    if 0 < RC < 1:
        # Calcular el valor z correspondiente al nivel de servicio
        z = stats.norm.ppf(RC)

        # Calcular y1
        y1 = z * sigma + D
        print(f"La cantidad economica de pedido es: {y1}")
```

Modelo con preparación (política s-S)

```
import sympy as sp

# Definición de constantes
D = sp.symbols('D')  # Definir D como símbolo
h = 2.39 # COSTO DE ALMACENAMIENTO
p = 10  #COSTO DE PENALIZACION
c = 8.13  #COSTO DE COMPRA
demanda = 28
K = 23.7  # COSTO DE PREPARACION DE PEDIDO
X = 16   # INVENTARIO INICIAL
```

```
S = f= ((p-c)/(p+h))*demanda

# Parte 1: Definir la variable y
y = sp.symbols('y')

# Calcular la integral de 0 a y de (y - D) / 10
integral1 = sp.integrate((y - D) / demanda, (D, 0, y))

# Calcular la integral de y a 10 de (D - y) / 10
integral2 = sp.integrate((D - y) / demanda, (D, y, demanda))

# Parte 2 y Parte 3 multiplicadas por sus coeficientes
resultado1 = c * (y - 0) + h * integral1 + p * integral2

# Simplificar el resultado
resultado1_simplificado = sp.simplify(resultado1)
print("Resultado de la integral en forma cuadrática (Ecuación 1):", resultado1_simplificado)

# Parte 2: Definir la variable v
v = sp.symbols('v')

# Calcular la integral de 0 a v de (v - D) / 10
integral1_v = sp.integrate((v - D) / demanda, (D, 0, v))

# Calcular la integral de v a 10 de (D - v) / 10
integral2_v = sp.integrate((D - v) / demanda, (D, v, demanda))

# Parte 2 y Parte 3 multiplicadas por sus coeficientes para v
resultado2 = c * (v - 0) + h * integral1_v + p * integral2_v

# Simplificar el resultado
resultado2_simplificado = sp.simplify(resultado2)
print("Resultado de la integral en forma cuadrática (Ecuación 2):", resultado2_simplificado)

# Sumar K a la ecuación 2 y reemplazar v por S=8
resultado3 = resultado2_simplificado.subs(v, S) + K  # Reemplazar v por S y sumar K


# Igualar la ecuación 1 y la ecuación 3
ecuacion = sp.Eq(resultado1_simplificado, resultado3)

# Reorganizar para la forma cuadrática en y
forma_cuadratica = sp.collect(ecuacion.lhs - ecuacion.rhs, y)

# Extraer coeficientes de la forma ax^2 + bx + c = 0
coeficientes = sp.poly(forma_cuadratica, y).all_coeffs()
a, b, c = coeficientes[0], coeficientes[1], coeficientes[2]
```

```
# Resolver la ecuación cuadrática
soluciones_y = sp.solve(sp.Eq(a * y**2 + b * y + c, 0), y)

# Definir nombres para las soluciones
sol1 = soluciones_y[0]
sol2 = soluciones_y[1]

# Lógica para descartar soluciones mayores que K
valores_validos = [sol for sol in [sol1, sol2] if sol <= K]

# Lógica para manejar la solución restante
if not valores_validos:
    resultado_final = "NO HAY SOLUCIONES VÁLIDAS"
else:
    solucion_restante = valores_validos[0]  # Tomar la solución restante
    if solucion_restante < 0:
        resultado_final = "NO REALIZAR PEDIDO"
    elif solucion_restante < X:
        cantidad_a_pedir = S - X
        resultado_final = f"LA CANTIDAD A PEDIR ES: {cantidad_a_pedir}"
    else:
        resultado_final = "NO REALIZAR PEDIDO"

# Mostrar el resultado final
print(resultado_final)
```

Modelo de varios periodos

```
# Modelos de varios periodos

# Definimos los parámetros
x = 100 #Inventario inicial
D= 10  #Demanda mas alta del intervalo
p = 3    #costo de penalizacion
r = 2    # ingreso por unidad
c = 1    # costo de compra (o de produccion) por unidad
h = 0.1  # costo de almacenamiento
alpha = 0.8  # factor de descuento por periodo

# Calculamos el valor
numerador = p + (1 + alpha) * (r - c)
denominador = p + h + (1 - alpha) * r
resultado_integral = numerador / denominador

resultado_integral * D
```

```
print(f"y*= {resultado_integral * D :.2f}")

# Decisión de pedido
if x < resultado_integral * D:
    pedido = resultado_integral * D - x

    print(f"Se debe pedir {pedido} unidades.")
else:
    print("No es necesario realizar un pedido.")
```

Anexo 6. Código unificado modelos determinísticos

```
#MODELO DE GESTION DE INVENTARIOS MODELOS DETERMINISTICOS#

import math
import numpy as np
import matplotlib.pyplot as plt
from sklearn.tree import DecisionTreeClassifier
from sklearn import tree

# Funciones originales

# Función para calcular la cantidad económica de pedido (EOQ)
def calcular_eoq(K, D, h):
    return math.sqrt((2 * K * D) / h)

# Función para calcular el costo total mensual
def calcular_costo_total_mensual(K, D, h, Q):
    costo_total_mensual = K / ((math.sqrt((2 * K * D) / h)) / D) + h * (math.sqrt((2 * K * D)
/ h)) / 2
    return costo_total_mensual

# Función para calcular el punto de reorden
def calcular_punto_reorden(D, L, K, h):
    to = (math.sqrt((2 * K * D) / h)) / D
    n = math.floor(L / to)
    Le = L - n * to
    LeD = Le * D
    return LeD

# Modelo de cantidad económica de pedido con descuentos por cantidad (nuevo código)
def modelo_inv_disc_v2(D, k, c1, c2, h, q, L):
```

```
    y_punto = ((2 * k * D) / h) ** 0.5

    # Cálculo del punto de reorden con la misma fórmula
    to = (math.sqrt((2 * k * D) / h)) / D
    n = math.floor(L / to)
    Le = L - n * to
    y_r = Le * D

    # Cálculo del TCU
    TCU = D * c1 + k * D / y_punto + (h * y_punto / 2)

    # Resolver la ecuación cuadrática
    def solve_quadratic(a, b, c, y_punto):
        discriminant = b ** 2 - 4 * c
        if discriminant < 0:
            return "No hay soluciones reales"
        x1 = (-b + math.sqrt(discriminant)) / 2
        x2 = (-b - math.sqrt(discriminant)) / 2
        if x1 > y_punto and x2 > y_punto:
            return max(x1, x2)
        elif x1 > y_punto:
            return x1
        elif x2 > y_punto:
            return x2
        else:
            return "No hay solución mayor que el umbral"

    # Parámetros de la ecuación cuadrática
    f = 2  # Coeficiente de x^2
    b = ((L * (c2 * D - TCU)) / h)
    c = (L * k * D / h)

    resultado = solve_quadratic(f, b, c, y_punto)

    if q > y_punto and q < resultado:
        y_opt = q
    else:
        y_opt = y_punto

    return y_opt, y_r

# Modelo de cantidad económica de pedido de varios artículos con limitación de almacén
def inv_restriccion_area(k, D, h, a, A, n):
    y_sub = []
    for i in range(n):
        yi = ((2 * k[i] * D[i]) / (h[i])) ** 0.5
```

```
        y_sub.append(yi)

    area_ocupada = np.dot(a, y_sub)

    if area_ocupada <= A:
        print("Se ha encontrado la sol óptima con y_punto")
        return y_sub, area_ocupada
    else:
        print("Calcular con Lagrange")
        diferencia = 10
        lamb = 0

        # Estableciendo el margen de error en 0.5
        error = 0.5

        while diferencia > error:
            y_opt = []
            for j in range(n):
                y_optimo = ((2 * k[j] * D[j]) / (h[j] - 2 * lamb * a[j])) ** 0.5
                y_opt.append(y_optimo)

            area_ocu_op = np.dot(a, y_opt)
            if area_ocu_op <= A:
                diferencia = np.abs(A - area_ocu_op)
                lamb -= 0.001
            lamb -= 0.001

        print("La solución óptima es:")
        return y_opt, area_ocu_op

# Modelo dinámico sin costo de preparación
def modelo_dinamico_sin_preparacion(demand, normal_capacity, overtime_capacity, normal_cost,
overtime_cost, storage_cost):
    num_periods = len(demand)
    total_cost = 0
    production_schedule = np.zeros((num_periods, 2), dtype=int)  # [Normal, Overtime]
production
    stored_units = np.zeros(num_periods, dtype=int)

    for i in range(num_periods):
        if i == 0:
            production_schedule[i, 0] = normal_capacity[i]
            remaining_demand = demand[i] - normal_capacity[i]
            if remaining_demand > 0:
                production_schedule[i, 1] = remaining_demand
            stored_units[i] = normal_capacity[i] + production_schedule[i, 1] - demand[i]
        else:
```

```
            if stored_units[i-1] < demand[i]:
                required_production = demand[i] - stored_units[i-1]
                production_schedule[i, 0] = min(required_production, normal_capacity[i])
                remaining_demand = required_production - production_schedule[i, 0]
                if remaining_demand > 0:
                    production_schedule[i, 1] = remaining_demand
            stored_units[i] = production_schedule[i, 0] + production_schedule[i, 1] +
stored_units[i-1] - demand[i]

        normal_production_cost = production_schedule[i, 0] * normal_cost
        overtime_production_cost = production_schedule[i, 1] * overtime_cost
        storage_cost_total = stored_units[i] * storage_cost
        total_cost += normal_production_cost + overtime_production_cost + storage_cost_total

    return production_schedule, total_cost
import numpy as np

# Modelo dinámico con costo de preparación
def modelo_dinamico_con_preparacion(D, K, h, x_inicial, c_pro1, c_pro2, unidades_min):
    max = sum(D)
    n = len(D)
    f = np.full((n + 1, max), np.inf)
    x_opt = np.zeros((n + 1, max), dtype=int)
    f[n, :] = 0  # Costo mínimo en el último periodo es 0

    def c_i(p_i):
        if p_i <= unidades_min:
            return c_pro1 * p_i
        else:
            return (c_pro1 + c_pro2) + c_pro2 * (p_i - unidades_min)

    def C_i(p_i, i):
        if p_i == 0:
            return 0
        else:
            return K[i] + c_i(p_i)

    for i in range(n - 1, -1, -1):
        for x in range(max):
            for p in range(max):
                x_next = x + p - D[i]
                if 0 <= x_next < max:
                    costo = C_i(p, i) + h[i] * x + f[i + 1, x_next]
                    if costo < f[i, x]:
                        f[i, x] = costo
                        x_opt[i, x] = p
```

```
    p_opt = []
    x = x_inicial
    for i in range(n):
        p_opt.append(x_opt[i, x])
        x = x + x_opt[i, x] - D[i]

    return p_opt, f[0, x_inicial]
# Interfaz para seleccionar el modelo y luego solicitar parámetros
def interfaz_usuario():
    while True:  # Bucle para permitir volver a calcular
        print("GESTION DE INVENTARIOS MODELOS DETERMINISTICOS")
        print("Selecciona el modelo que deseas calcular:")
        print("1: Cantidad económica de pedido (EOQ)")
        print("2: Modelo con Descuentos por Cantidad")
        print("3: Modelo de Cantidad Económica de Pedido de Varios Artículos con Restricción
de Almacén")
        print("4: Modelo Dinámico Sin Costo de Preparación")
        print("5: Modelo Dinámico Con Costo de Preparación")

        decision = input("Introduce el número del modelo: ")

        if decision == "1":
            print("Parámetros requeridos para EOQ:")
            D = float(input("Demanda mensual (D): "))
            K = float(input("Costo de pedido (K): "))
            h = float(input("Costo de mantenimiento por unidad por mes (h): "))
            L = float(input("Tiempo de entrega en meses (L): "))

            eoq = calcular_eoq(K, D, h)
            costo_total = calcular_costo_total_mensual(K, D, h, eoq)
            punto_reorden = calcular_punto_reorden(D, L, K, h)

            print(f"Cantidad económica de pedido (EOQ): {eoq:.2f}")
            print(f"Costo total mensual: {costo_total:.2f}")
            print(f"Punto de reorden: {punto_reorden:.2f}")

        elif decision == "2":
            print("Parámetros requeridos para el Modelo con Descuentos por Cantidad:")
            K = float(input("Costo de pedido (K): "))
            D = float(input("Demanda mensual (D): "))
            h = float(input("Costo de mantenimiento por unidad por mes (h): "))
            c1 = float(input("Costo por unidad sin descuento (c1): "))
            c2 = float(input("Costo por unidad con descuento (c2): "))
            q = float(input("Cantidad a pedir (q): "))
            L = float(input("Tiempo de entrega en meses (L): "))

            y_opt, y_r = modelo_inv_disc_v2(D, K, c1, c2, h, q, L)
```

```
            print(f"Cantidad óptima a pedir: {y_opt:.2f}")
            print(f"Punto de reorden: {y_r:.2f}")

        elif decision == "3":
            print("Parámetros requeridos para el Modelo de Cantidad Económica de Pedido de Varios Artículos con Restricción de Almacén:")
            n = int(input("Número de artículos: "))
            k = []
            D = []
            h = []
            a = []
            for i in range(n):
                print(f"Artículo {i + 1}:")
                k.append(float(input("Costo de pedido (K): ")))
                D.append(float(input("Demanda mensual (D): ")))
                h.append(float(input("Costo de mantenimiento por unidad por mes (h): ")))
                a.append(float(input("Área ocupada por unidad (a): ")))

            A = float(input("Área total disponible (A): "))

            y_sub, area_ocupada = inv_restriccion_area(k, D, h, a, A, n)
            print(f"Cantidad óptima para cada artículo: {y_sub}")
            print(f"Área ocupada: {area_ocupada:.2f}")

        elif decision == "4":
            print("Parámetros requeridos para el Modelo Dinámico Sin Costo de Preparación:")
            demand = list(map(int, input("Demanda por período (separados por espacios): ").split()))
            normal_capacity = list(map(int, input("Capacidad normal por período (separados por espacios): ").split()))
            overtime_capacity = list(map(int, input("Capacidad de horas extras por período (separados por espacios): ").split()))
            normal_cost = float(input("Costo de producción normal por unidad: "))
            overtime_cost = float(input("Costo de producción en horas extras por unidad: "))
            storage_cost = float(input("Costo de almacenamiento por unidad por período: "))

            production_schedule, total_cost = modelo_dinamico_sin_preparacion(demand, normal_capacity, overtime_capacity, normal_cost, overtime_cost, storage_cost)
            print(f"Programa de producción:\n{production_schedule}")
            print(f"Costo total: {total_cost:.2f}")

        elif decision == "5":
            print("Parámetros requeridos para el Modelo Dinámico Con Costo de Preparación:")
            D = list(map(int, input("Demanda por período (separados por espacios): ").split()))
            K = list(map(float, input("Costo de pedido por período (separados por espacios): ").split()))
            h = list(map(float, input("Costo de mantenimiento por período (separados por espacios): ").split()))
```

```
            x_inicial = int(input("Unidades iniciales en inventario: "))
            c_pro1 = float(input("Costo de producción por unidad (c_pro1): "))
            c_pro2 = float(input("Costo adicional por producción adicional (c_pro2): "))
            unidades_min = int(input("Cantidad mínima a producir: "))

            p_opt, costo_total = modelo_dinamico_con_preparacion(D, K, h, x_inicial, c_pro1,
c_pro2, unidades_min)
            print(f"Producción óptima por período: {p_opt}")
            print(f"Costo total: {costo_total:.2f}")

        else:
            print("Opción no válida. Por favor, selecciona un número del 1 al 5.")

        # Pregunta si desea calcular otro modelo
        volver_a_calcular = input("¿Deseas realizar el cálculo de otro modelo? (sí = 1 / no =
0): ")
        if volver_a_calcular != "1":
            print("Gracias por usar el programa.")
            break
# Ejecutar la interfaz de usuario
if __name__ == "__main__":
    interfaz_usuario()
```

Anexo 7. Código unificado modelos probabilísticos

```
import scipy.stats as stats
import math
import sympy as sp
import numpy as np

def calcular_eoq_probabilizado(D, sigma, L, alpha):
    mu_L = D * L
    sigma_L = sigma * math.sqrt(L)
    Z = stats.norm.ppf(1 - alpha)
    B = Z * sigma_L
    R = mu_L + B
    return R, B

def modelo_probabilistico(K, h, p, D, fx, denominador_fx, Ex):
    y1 = math.sqrt(2 * D * (K + p * Ex) / h)
    y2 = (p * D) / h

    if y2 >= y1:
        x, R, v = sp.symbols('x R v')
        limite_inferior = R
```

```
S = sp.integrate((x - R) * fx, (x, limite_inferior, denominador_fx))
S_simplificada = sp.simplify(S)
yi = sp.sqrt((2 * D * (K + p * v)) / h)
yi1 = sp.symbols('yi1')

z = denominador_fx
integral_result = sp.integrate(fx, (x, R, z))
ecuacion = sp.Eq(integral_result, (h * yi1) / (p * D))
R_resultado = sp.solve(ecuacion, R)[0]
yn1 = math.sqrt(2 * K * D / h)
nueva_ecuacion = ecuacion.subs(yi1, yn1)
nuevo_R = sp.solve(nueva_ecuacion, R)[0]

S_final = S_simplificada.subs(R, nuevo_R)
y_it1 = yi.subs(v, S_final)
nueva_ecuacion2 = ecuacion.subs(yi1, y_it1)
nuevo_R2 = sp.solve(nueva_ecuacion2, R)[0]
S_final2 = S_simplificada.subs(R, nuevo_R2)
y_it2 = yi.subs(v, S_final2)
nueva_ecuacion3 = ecuacion.subs(yi1, y_it2)
nuevo_R3 = sp.solve(nueva_ecuacion3, R)[0]
S_final3 = S_simplificada.subs(R, nuevo_R3)
y_it3 = yi.subs(v, S_final3)
nueva_ecuacion4 = ecuacion.subs(yi1, y_it3)
nuevo_R4 = sp.solve(nueva_ecuacion4, R)[0]
S_final4 = S_simplificada.subs(R, nuevo_R4)
y_it4 = yi.subs(v, S_final4)
nueva_ecuacion5 = ecuacion.subs(yi1, y_it4)
nuevo_R5 = sp.solve(nueva_ecuacion5, R)[0]
S_final5 = S_simplificada.subs(R, nuevo_R5)
y_it5 = yi.subs(v, S_final5)

print(f"yi_1: {y_it5}")
nueva_ecuacion6 = ecuacion.subs(yi1, y_it5)
nuevo_R6 = sp.solve(nueva_ecuacion6, R)[0]
print(f"Ri_1 = {nuevo_R6}")

S_final6 = S_simplificada.subs(R, nuevo_R6)
y_it6 = yi.subs(v, S_final6)
print(f"yi_2: {y_it6}")

nueva_ecuacion7 = ecuacion.subs(yi1, y_it6)
nuevo_R7 = sp.solve(nueva_ecuacion7, R)[0]
print(f"Ri_2 = {nuevo_R7}")

print("")
```

```
        print("POLITICA OPTIMA DE PEDIDO")
        print(f"Se debe pedir aproximadamente {y_it6:2f} unidades, siempre que el nivel de existencias baje a {nuevo_R7:2f} ")
    else:
        print("No cumplió la restricción para seguir con el cálculo.")


def modelo_probabilistico_sin_costo_de_preparacion(c, h, p, D, sigma):
    if p < c:
        print("No se puede continuar el cálculo porque el costo de compra del artículo es mayor que la penalización por no suministrarlo.")
    else:
        RC = (p - c) / (p + h)
        if 0 < RC < 1:
            z = stats.norm.ppf(RC)
            y1 = z * sigma + D
            print(f"La cantidad economica de pedido es: {y1}")


def modelo_probabilistico_sin_costo_de_preparacion_discreta(c, h, p, Distrib_discreta, FD):
    if p < c:
        print("No se puede continuar el cálculo porque el costo de compra del artículo es mayor que la penalización por no suministrarlo.")
    else:
        RC = (p - c) / (p + h)


        def calcular_distribucion_acumulada(valores, probabilidades, RC):
            if not np.isclose(sum(probabilidades), 1):
                raise ValueError("Las probabilidades deben sumar 1.")
            distribucion_acumulada = np.cumsum(probabilidades)
            for i, acumulada in enumerate(distribucion_acumulada):
                if RC <= acumulada:
                    return valores[i]
            return None


        valor_mas_alto = calcular_distribucion_acumulada(Distrib_discreta, FD, RC)
        if valor_mas_alto is not None:
            print(f"De acuerdo a la funcion de distribucion de probabilidades discreta la cantidad de pedido es: {valor_mas_alto}")
        else:
            print(f"RC = {RC} es mayor que la acumulada total.")


def modelo_varios_periodos(x, D, p, r, c, h, alpha):
    numerador = p + (1 + alpha) * (r - c)
    denominador = p + h + (1 - alpha) * r
    resultado_integral = numerador / denominador
    resultado = resultado_integral * D

    print(f"y*: {resultado:.2f}")
```

```
    if x < resultado:
        pedido = resultado - x
        print(f"Se debe pedir {pedido} unidades.")
    else:
        print("No es necesario realizar un pedido.")

def modelo_s_s(h, p, c, demanda, K, X):
    # Definición de constantes
    D = sp.symbols('D')  # Definir D como símbolo

    # Calcular S
    S = ((p - c) / (p + h)) * demanda

    # Parte 1: Definir la variable y
    y = sp.symbols('y')

    # Calcular la integral de 0 a y de (y - D) / 10
    integral1 = sp.integrate((y - D) / demanda, (D, 0, y))

    # Calcular la integral de y a 10 de (D - y) / 10
    integral2 = sp.integrate((D - y) / demanda, (D, y, demanda))

    # Parte 2 y Parte 3 multiplicadas por sus coeficientes
    resultado1 = c * (y - 0) + h * integral1 + p * integral2

    # Simplificar el resultado
    resultado1_simplificado = sp.simplify(resultado1)
    print("Resultado de la integral en forma cuadrática (Ecuación 1):",
resultado1_simplificado)

    # Parte 2: Definir la variable v
    v = sp.symbols('v')

    # Calcular la integral de 0 a v de (v - D) / 10
    integral1_v = sp.integrate((v - D) / demanda, (D, 0, v))

    # Calcular la integral de v a 10 de (D - v) / 10
    integral2_v = sp.integrate((D - v) / demanda, (D, v, demanda))

    # Parte 2 y Parte 3 multiplicadas por sus coeficientes para v
    resultado2 = c * (v - 0) + h * integral1_v + p * integral2_v

    # Simplificar el resultado
    resultado2_simplificado = sp.simplify(resultado2)
    print("Resultado de la integral en forma cuadrática (Ecuación 2):",
resultado2_simplificado)
```

```
        # Sumar K a la ecuación 2 y reemplazar v por S
        resultado3 = resultado2_simplificado.subs(v, S) + K  # Reemplazar v por S y sumar K

        # Igualar la ecuación 1 y la ecuación 3
        ecuacion = sp.Eq(resultado1_simplificado, resultado3)

        # Reorganizar para la forma cuadrática en y
        forma_cuadratica = sp.collect(ecuacion.lhs - ecuacion.rhs, y)

        # Extraer coeficientes de la forma ax^2 + bx + c = 0
        coeficientes = sp.poly(forma_cuadratica, y).all_coeffs()
        a, b, c = coeficientes[0], coeficientes[1], coeficientes[2]

        # Resolver la ecuación cuadrática
        soluciones_y = sp.solve(sp.Eq(a * y**2 + b * y + c, 0), y)

        # Definir nombres para las soluciones
        sol1 = soluciones_y[0]
        sol2 = soluciones_y[1]

        # Lógica para descartar soluciones mayores que K
        valores_validos = [sol for sol in [sol1, sol2] if sol <= K]

        # Lógica para manejar la solución restante
        if not valores_validos:
            resultado_final = "NO HAY SOLUCIONES VÁLIDAS"
        else:
            solucion_restante = valores_validos[0]  # Tomar la solución restante
            if solucion_restante < 0:
                resultado_final = "NO REALIZAR PEDIDO"
            elif solucion_restante < X:
                cantidad_a_pedir = S - X
                resultado_final = f"LA CANTIDAD A PEDIR ES: {cantidad_a_pedir}"
            else:
                resultado_final = "NO REALIZAR PEDIDO"

        # Mostrar el resultado final
        print(resultado_final)

def main():
    while True:
        print("GESTION DE INVENTARIOS PROBABILISTICOS")
        print("Seleccione el modelo que desea calcular:")
        print("1. Modelo 'probabilizado' de cantidad económica de pedido")
        print("2. Modelo probabilístico de cantidad económica de pedido")
        print("3. Modelo probabilístico sin costo de preparación (distribución continua)")
```

```
    print("4. Modelo probabilístico sin costo de preparación (distribución discreta)")
    print("5. Modelo de varios periodos")
    print("6. Modelo con politica S-S")

    modelo = int(input("Ingrese el número del modelo (1, 2, 3, 4, 5 o 6): "))

    if modelo == 1:
        D = float(input("Ingrese la demanda (D): "))
        sigma = float(input("Ingrese la desviación estándar de la demanda (sigma): "))
        L = float(input("Ingrese el tiempo de entrega (L): "))
        alpha = float(input("Ingrese la probabilidad máxima admisible de agotamiento de
existencias (alpha): "))

        R, B = calcular_eoq_probabilizado(D, sigma, L, alpha)
        print(f"Punto de reorden (R): {R:.2f} unidades")
        print(f"Stock de seguridad (B): {B:.2f} unidades")

    elif modelo == 2:
        K = float(input("Ingrese el costo de preparación de pedido (K): "))
        h = float(input("Ingrese el costo de almacenamiento (h): "))
        p = float(input("Ingrese el costo de faltante por unidad de inventario (p): "))
        D = float(input("Ingrese la demanda (D): "))
        fx = eval(input("Ingrese el valor de fx (por ejemplo 1/100): "))  # Usar eval para
convertir la expresión
        denominador_fx = float(input("Ingrese el denominador de fx: "))
        Ex = float(input("Ingrese el valor esperado (Ex): "))

        modelo_probabilistico(K, h, p, D, fx, denominador_fx, Ex)

    elif modelo == 3:
        c = float(input("Ingrese el costo de compra (c): "))
        h = float(input("Ingrese el costo de almacenamiento (h): "))
        p = float(input("Ingrese el costo de penalización (p): "))
        D = float(input("Ingrese la demanda promedio (D): "))
        sigma = float(input("Ingrese la desviación estándar de la demanda (sigma): "))

        modelo_probabilistico_sin_costo_de_preparacion(c, h, p, D, sigma)

    elif modelo == 4:
        c = float(input("Ingrese el costo de compra (c): "))
        h = float(input("Ingrese el costo de almacenamiento (h): "))
        p = float(input("Ingrese el costo de penalización (p): "))

        # Introducir valores y probabilidades para la distribución discreta
        Distrib_discreta = list(map(int, input("Ingrese los valores de la variable aleatoria
separados por comas: ").split(',')))
        FD = list(map(float, input("Ingrese las probabilidades asociadas a cada valor
separados por comas: ").split(',')))
```

```
            modelo_probabilistico_sin_costo_de_preparacion_discreta(c, h, p, Distrib_discreta, FD)

        elif modelo == 5:
            x = float(input("Ingrese el inventario inicial (x): "))
            D = float(input("Ingrese la demanda más alta del intervalo (D): "))
            p = float(input("Ingrese el costo de penalización (p): "))
            r = float(input("Ingrese el ingreso por unidad (r): "))
            c = float(input("Ingrese el costo de compra (o de producción) por unidad (c): "))
            h = float(input("Ingrese el costo de almacenamiento (h): "))
            alpha = float(input("Ingrese el factor de descuento por periodo (alpha): "))

            modelo_varios_periodos(x, D, p, r, c, h, alpha)

        elif modelo == 6:
          # Ingresar parámetros para el modelo S-s
            h = float(input("Ingrese el costo de almacenamiento (h): "))
            p = float(input("Ingrese el costo de penalización (p): "))
            c = float(input("Ingrese el costo de compra (c): "))
            demanda = float(input("Ingrese la demanda: "))
            K = float(input("Ingrese el costo de preparación de pedido (K): "))
            X = float(input("Ingrese el inventario inicial (X): "))

            modelo_s_s(h, p, c, demanda, K, X)  # Llamar al modelo S-s con los parámetros
ingresados

        else:
            print("Opción no válida.")

        # Pregunta si desea calcular otro modelo
        volver_a_calcular = input("¿Deseas realizar el cálculo de otro modelo? (sí = 1 / no = 0):
")
        if volver_a_calcular != "1":
           print("Gracias por usar el programa.")
           break

if __name__ == "__main__":
    main()
```

Anexo 8. Modelos determinísticos

Modelo clásico de cantidad económica de pedido

SIMULACIÓN DE DATOS

Modelo clásico de cantidad económica de pedido

		Tasa de demanda	Costo de preparación de pedido	Costo unitario de almacenamiento	Tiempo de entrega	CANTIDAD DE PEDIDO	PUNTO DE REORDEN
1	**Parámetros**	100	100	0,02	12	1000	200
2	**Parámetros**	300	20	0,21	1	239	60
3	**Parámetros**	120	30	0,04	5	424	175
4	**Parámetros**	145	23	0,05	6	365	139
5	**Parámetros**	80	15	0,03	10	282	234
6	**Parámetros**	3000	200	0,4	20	1732	1110
7	**Parámetros**					489	220
		240	30	0,06	5		
8	**Parámetros**	450	50	0,3	7	387	51
9	**Parámetros**	134	35	0,05	4	433	102
10	**Parámetros**	222	42	0,1	6	431	36
11	**Parámetros**	345	15	0,3	5	185	53
12	**Parámetros**	500	21	0,04	3	724	50
13	**Parámetros**	1200	40	0,1	5	979	121
14	**Parámetros**	1340	41	0,1	1	1048	291
15	**Parámetros**	340	12	0,04	4	451	5
16	**Parámetros**	120	5	0,01	6	346	27
17	**Parámetros**	340	14	0,02	5	689	320
18	**Parámetros**	524	42,16	1,55	1	168	17

SIMULACIÓN DE DATOS

Modelo clásico de cantidad económica de pedido

		Tasa de demanda	Costo de preparación de pedido	Costo unitario de almacenamiento	Tiempo de entrega	CANTIDAD DE PEDIDO	PUNTO DE REORDEN
19	**Parámetros**	832	69,82	2,96	5	198	197
20	**Parámetros**	234	117,63	7,89	9	83	17
21	**Parámetros**	450	18,49	3,47	4	69	68
22	**Parámetros**	913	86,12	0,82	2	437	74
23	**Parámetros**	162	34,71	4,01	7	52	21
24	**Parámetros**	775	121,93	2,24	12	290	5
25	**Parámetros**	610	11,38	5,76	6	49	26
26	**Parámetros**	218	53,29	9,34	11	49	3
27	**Parámetros**	435	93,47	8,58	10	97	66
28	**Parámetros**	743	27,24	0,14	8	534	527
29	**Parámetros**	322	65,98	7,45	3	75	59
30	**Parámetros**	587	76,31	6,87	4	114	64
31	**Parámetros**	158	34,85	3,94	6	52	49
32	**Parámetros**	901	14,63	1,13	5	152	75
33	**Parámetros**	278	52,17	4,21	2	83	57
34	**Parámetros**	493	91,08	8,77	8	101	98
35	**Parámetros**	656	24,31	5,5	7	76	22

SIMULACIÓN DE DATOS

Modelo clásico de cantidad económica de pedido

		Tasa de demanda	Costo de preparación de pedido	Costo unitario de almacenamiento	Tiempo de entrega	CANTIDAD DE PEDIDO	PUNTO DE REORDEN
36	**Parámetros**	317	41,25	9,59	9	52	31
37	**Parámetros**	832	61,53	2,19	4	216	84
38	**Parámetros**	510	88,26	6,84	11	114	103
39	**Parámetros**	190	35,42	7,08	3	43	3
40	**Parámetros**	764	72,34	0,29	1	617	146
41	**Parámetros**	289	60,71	4,52	10	88	70
42	**Parámetros**	580	37,24	5,99	8	84	54
43	**Parámetros**	423	19,82	5,99	5	52	51
44	**Parámetros**	640	102,31	1,86	6	265	125
45	**Parámetros**	173	47,49	2,98	3	74	73
46	**Parámetros**	915	89,35	6,03	2	164	18
47	**Parámetros**	328	54,29	4,78	7	86	51
48	**Parámetros**	689	32,96	9,21	12	70	51
49	**Parámetros**	142	100	0,09	4	516	6
50	**Parámetros**	514	300	0,1	1	1756	514
51	**Parámetros**	835	16,29	1,12	5	155	122
52	**Parámetros**	267	53,42	3,88	9	85	2

SIMULACIÓN DE DATOS

Modelo clásico de cantidad económica de pedido

		Tasa de demanda	Costo de preparación de pedido	Costo unitario de almacenamiento	Tiempo de entrega	CANTIDAD DE PEDIDO	PUNTO DE REORDEN
53	**Parámetros**	401	352	0,06	3	2169	1203
54	**Parámetros**	751	698	0,06	4	4180	3004
55	**Parámetros**	290	142	0,04	8	1434	885
56	**Parámetros**	563	489	0,03	6	4284	3378
57	**Parámetros**	476	200	0,19	11	1001	230
58	**Parámetros**	829	598	0,02	7	7040	5803
59	**Parámetros**	341	256	0,05	2	1868	682
60	**Parámetros**	635	548	0,05	4	3730	2540
61	**Parámetros**	197	100	0,6	5	256	216
62	**Parámetros**	720	565	0,04	9	4509	1970
63	**Parámetros**	284	71,34	3,25	1	111	60
64	**Parámetros**	594	59,47	7,36	6	97	36
65	**Parámetros**	460	17,13	0,57	3	166	49
66	**Parámetros**	732	82,9	4,88	8	157	21
67	**Parámetros**					44	42
		130	20,35	2,64	11		
68	**Parámetros**	568	36,14	5,93	2	83	54
69	**Parámetros**	293	48,62	1,43	4	141	42

SIMULACIÓN DE DATOS

Modelo clásico de cantidad económica de pedido

		Tasa de demanda	Costo de preparación de pedido	Costo unitario de almacenamiento	Tiempo de entrega	CANTIDAD DE PEDIDO	PUNTO DE REORDEN
70	**Parámetros**	650	13,91	8,07	5	47	31
71	**Parámetros**	210	87,73	6,61	6	74	65
72	**Parámetros**	457	27,4	7,89	1	56	6
73	**Parámetros**	891	14,52	0,89	3	170	115
74	**Parámetros**	369	58,73	9,57	10	67	55
75	**Parámetros**	713	49,67	3,03	5	152	48
76	**Parámetros**	194	82,13	8,96	7	59	46
77	**Parámetros**	530	39,84	4,62	9	95	85
78	**Parámetros**	402	65,17	7,25	2	85	38
79	**Parámetros**	874	53,69	6,18	4	123	45
80	**Parámetros**	119	22,34	1,95	6	52	35
81	**Parámetros**	578	77,56	8,72	11	101	71
82	**Parámetros**	312	48,12	2,47	5	110	16
83	**Parámetros**	723	29,17	9,35	1	67	51
84	**Parámetros**	150	63,29	4,4	8	65	17
85	**Parámetros**	607	19,78	6	3	63	49
86	**Parámetros**	276	58,45	7,63	10	65	28

SIMULACIÓN DE DATOS

Modelo clásico de cantidad económica de pedido

		Tasa de demanda	Costo de preparación de pedido	Costo unitario de almacenamiento	Tiempo de entrega	CANTIDAD DE PEDIDO	PUNTO DE REORDEN
87	**Parámetros**	731	30,81	8,09	6	74	58
88	**Parámetros**	231	120,98	0,34	9	405	51
89	**Parámetros**	370	104,25	2,15	8	189	118
90	**Parámetros**	425	15,5	3,86	12	58	17
91	**Parámetros**	167	158,4	9,47	4	74	70
92	**Parámetros**	648	172,5	1,23	7	426	272
93	**Parámetros**	350	5,9	0,98	11	64	19
94	**Parámetros**	234	43,73	2,61	10	88	37
95	**Parámetros**	500	67,8	4,55	3	122	35
96	**Parámetros**	128	149,5	0,06	8	798	225
97	**Parámetros**	314	118,2	6,31	1	108	97
98	**Parámetros**	910	19,6	3,7	5	98	33
99	**Parámetros**	680	79,2	0,25	2	656	47
100	**Parámetros**	150	144,3	0,09	4	693	600

Cantidad económica de pedido con discontinuidades de precio

SIMULACIÓN DE DATOS

Modelo clásico de cantidad económica de pedido con discontinuidades de precio

									CANTIDAD DE PEDIDO	PUNTO DE REORDEN
		Tasa de demanda	Costo de preparación de pedido	Costo compra normal	Costo compra con descuento	Costo unitario/ u. Tiempo de almacenamiento	Tamaño mínimo del pedido con descuento	Tiempo de entrega		
1	**Parámetros**	396	505	11,44	9,25	0,03	875	22	3778	1154
2	**Parámetros**	811	230	17,6	9,63	0,43	723	11	931	535
3	**Parámetros**	822	57	13,06	6,86	0,47	207	2	446	305
4	**Parámetros**	59	552	19,49	4	0,14	513	20	688	491
5	**Parámetros**	519	230	14,54	1,83	0,42	118	8	753	384
6	**Parámetros**	866	488	15,25	1,28	0,43	378	29	1395	1393
7	**Parámetros**	883	331	19,14	7,5	0,33	741	10	1321	901
8	**Parámetros**	932	56	16,83	6,88	0,01	326	10	2719	1161
9	**Parámetros**	891	85	19,38	1,09	0,48	375	3	563	420
10	**Parámetros**	311	535	17	3,31	0,12	163	17	1642	359
11	**Parámetros**	213	513	11,39	2,35	0,35	719	22	788	745
12	**Parámetros**	104	517	19,59	8,4	0,31	828	15	828	387
13	**Parámetros**	168	236	14,08	3,95	0,15	222	29	722	534
14	**Parámetros**	650	124	16,12	5,48	0,27	935	15	935	481
15	**Parámetros**	170	286	15,68	2,35	0,09	874	29	1063	675
16	**Parámetros**	311	592	12,14	9,01	0,4	993	30	993	732
17	**Parámetros**	667	499	19,08	4,91	0,06	243	5	3285	49
18	**Parámetros**	974	113	16,26	7,28	0,39	532	4	750	145
19	**Parámetros**	127	380	16,02	4,15	0,46	741	9	741	228

Modelo clásico de cantidad económica de pedido con discontinuidades de precio

									CANTIDAD DE PEDIDO	PUNTO DE REORDEN
		Tasa de demanda	Costo de preparación de pedido	Costo compra normal	Costo compra con descuento	Costo unitario/ u. Tiempo de almacenamiento	Tamaño mínimo del pedido con descuento	Tiempo de entrega		
20	**Parámetros**	67	205	12,32	7,7	0,22	389	30	389	234
21	**Parámetros**	787	161	15,7	4,06	0,21	199	8	1089	847
22	**Parámetros**	736	201	15,05	6,1	0,19	903	4	1234	474
23	**Parámetros**	200	578	18,17	5,69	0,29	282	5	897	102
24	**Parámetros**	976	469	15,07	7,89	0,17	509	22	2334	457
25	**Parámetros**	48	142	11,24	8,1	0,46	363	23	363	71
26	**Parámetros**	88	457	11,47	5,6	0,32	899	2	899	176
27	**Parámetros**	905	203	19,4	6,47	0,37	910	7	989	395
28	**Parámetros**	920	556	14,76	9,65	0,04	146	8	4983	2376
29	**Parámetros**	156	368	16,79	6,11	0,48	866	10	866	100
30	**Parámetros**	348	488	14,02	8,76	0,17	350	13	1434	220
31	**Parámetros**	549	488	11	5,81	0,21	493	24	1610	295
32	**Parámetros**	173	89	12,86	2,1	0,08	589	19	635	107
33	**Parámetros**	447	267	19,33	8,09	0,18	183	13	1142	100
34	**Parámetros**	628	377	17,03	1,8	0,21	278	3	1510	373
35	**Parámetros**	902	118	16,9	9,55	0,35	898	13	898	52
36	**Parámetros**	20	509	13,06	8,58	0,46	505	18	505	148
37	**Parámetros**	320	394	18,54	9,42	0,08	769	3	1786	960
38	**Parámetros**	662	217	13,68	2,13	0,11	473	2	1603	1324
39	**Parámetros**	468	536	14,17	4,98	0,33	775	30	1239	410

Modelo clásico de cantidad económica de pedido con discontinuidades de precio

		Tasa de demanda	Costo de preparación de pedido	Costo compra normal	Costo compra con descuento	Costo unitario/ u. Tiempo de almacenamiento	Tamaño mínimo del pedido con descuento	Tiempo de entrega	CANTIDAD DE PEDIDO	PUNTO DE REORDEN
40	**Parámetros**	215	423	16	3,19	0,38	860	20	860	150
41	**Parámetros**	766	553	16,5	2,21	0,38	555	8	1485	184
42	**Parámetros**	970	81	14,04	2,17	0,44	346	10	596	161
43	**Parámetros**	853	104	12,82	1,89	0,05	431	23	1959	23
44	**Parámetros**	879	321	15,09	6,02	0,18	163	22	1755	25
45	**Parámetros**	103	599	15,5	5,55	0,23	315	16	736	174
46	**Parámetros**	25	297	13,31	9,61	0,12	551	18	551	92
47	**Parámetros**	870	380	14,38	9,04	0,46	874	11	1194	10
48	**Parámetros**	228	239	16,95	4,09	0,07	275	23	1275	141
49	**Parámetros**	415	240	12,75	4,24	0,43	229	20	677	174
50	**Parámetros**	787	359	15,77	5,33	0,3	129	13	1376	595
51	**Parámetros**	846	507	14,44	6,86	0,28	395	27	1756	6
52	**Parámetros**	485	124	11,1	7,85	0,45	581	17	581	15
53	**Parámetros**	716	456	19,42	9,18	0,44	107	14	1213	312
54	**Parámetros**	702	377	14,9	8,56	0,45	123	26	1084	902
55	**Parámetros**	972	101	16,54	1,05	0,49	938	17	938	47
56	**Parámetros**	878	59	15,27	4,97	0,13	455	19	883	783
57	**Parámetros**	609	108	13,51	5,96	0,19	679	19	840	640
58	**Parámetros**	413	244	18,04	6,42	0,07	744	27	1689	1013
59	**Parámetros**	765	162	11,91	4	0,35	738	7	837	332

Modelo clásico de cantidad económica de pedido con discontinuidades de precio

		Tasa de demanda	Costo de preparación de pedido	Costo compra normal	Costo compra con descuento	Costo unitario/ u. Tiempo de almacenamiento	Tamaño mínimo del pedido con descuento	Tiempo de entrega	CANTIDAD DE PEDIDO	PUNTO DE REORDEN
60	**Parámetros**	935	281	16,69	2,33	0,05	687	3	3224	2805
61	**Parámetros**	580	473	15,53	5,27	0,11	690	24	2195	745
62	**Parámetros**	502	318	17,8	6,43	0,18	593	29	1350	1054
63	**Parámetros**	32	64	15,97	4,02	0,13	593	29	593	32
64	**Parámetros**	28	305	18,69	5,17	0,43	406	6	406	168
65	**Parámetros**	223	227	18,25	7,44	0,18	304	22	754	381
66	**Parámetros**	954	380	18,84	1,91	0,41	373	19	1336	751
67	**Parámetros**	928	245	12,43	2,29	0,11	456	15	2079	1444
68	**Parámetros**	962	447	11,01	3,34	0,04	548	15	4476	999
69	**Parámetros**	697	466	16,28	6,36	0,09	888	9	2636	999
70	**Parámetros**	895	490	19,53	5,41	0,43	397	2	1425	364
71	**Parámetros**	590	548	14,4	7	0,37	509	19	1320	646
72	**Parámetros**	580	350	13,17	9,19	0,37	461	24	1047	304
73	**Parámetros**	590	450	14,44	4,51	0,05	657	17	3356	3316
74	**Parámetros**	189	459	13,43	8,52	0,45	213	24	623	171
75	**Parámetros**	14	372	14,17	8,68	0,47	232	24	232	39
76	**Parámetros**	243	306	16,35	2,61	0,46	847	20	847	326
77	**Parámetros**	267	493	14,36	7,29	0,28	740	13	962	583
78	**Parámetros**	789	408	19,71	3,35	0,2	681	27	1816	1318
79	**Parámetros**	503	465	17,48	3,12	0,41	325	19	1072	977

Modelo clásico de cantidad económica de pedido con discontinuidades de precio

		Tasa de demanda	Costo de preparación de pedido	Costo compra normal	Costo compra con descuento	Costo unitario/ u. Tiempo de almacenamiento	Tamaño mínimo del pedido con descuento	Tiempo de entrega	CANTIDAD DE PEDIDO	PUNTO DE REORDEN
80	**Parámetros**	618	117	13,27	9,61	0,1	497	11	1175	923
81	**Parámetros**	929	54	19,31	8,71	0,4	580	9	580	353
82	**Parámetros**	677	101	18,03	2,1	0,33	379	1	640	36
83	**Parámetros**	520	463	14,57	6,16	0,12	263	22	2042	1227
84	**Parámetros**	879	75	19,62	3,44	0,18	353	22	851	606
85	**Parámetros**	251	560	15,83	3,77	0,46	724	25	779	36
86	**Parámetros**	698	405	15,26	2,32	0,37	387	14	1237	1109
87	**Parámetros**	988	332	11,85	3,02	0,27	466	14	1552	1414
88	**Parámetros**	511	347	11,57	4,4	0,36	836	16	989	263
89	**Parámetros**	997	546	13,28	9,26	0,17	515	17	2563	1570
90	**Parámetros**	533	373	11,85	2,46	0,45	226	8	941	496
91	**Parámetros**	284	469	12,26	8,42	0,39	544	10	830	349
92	**Parámetros**	616	344	18,74	3,54	0,28	523	15	1234	599
93	**Parámetros**	146	360	13,95	5,77	0,41	118	21	507	23
94	**Parámetros**	480	411	16,77	7,87	0,05	855	22	2695	2472
95	**Parámetros**	147	574	16,85	6,65	0,29	376	7	757	272
96	**Parámetros**	125	66	12,71	4,31	0,3	266	24	266	168
97	**Parámetros**	63	411	12,5	7,21	0,35	993	8	993	120
98	**Parámetros**	662	382	13,06	7,82	0,43	779	11	1087	754
99	**Parámetros**	861	533	18,09	8,03	0,08	298	16	3387	227

SIMULACIÓN DE DATOS

Modelo clásico de cantidad económica de pedido con discontinuidades de precio

		Tasa de demanda	Costo de preparación de pedido	Costo compra normal	Costo compra con descuento	Costo unitario/ u. Tiempo de almacenamiento	Tamaño mínimo del pedido con descuento	Tiempo de entrega	CANTIDAD DE PEDIDO	PUNTO DE REORDEN
100	**Parámetros**	443	147	17,7	1,71	0,12	201	5	1052	109

Cantidad económica de pedido de varios artículos con limitación de almacén

SIMULACIÓN DE DATOS

Modelo clásico de cantidad económica de varios artículos con limitación de almacén

Parámetros	Articulo N°	Tasa de demanda	Costo de preparación de pedido	Costo unitario/ u. Tiempo de almacenamiento	Área de almacenamiento por unidad	Área máxima disponible de almacenamiento para los n artículos	CANTIDAD DE PEDIDO	ÁREA TOTAL OCUPADA
1	1 2 3	2 4 4	10 5 15	0,3 0,1 0,2	1 1 1	25	6,34 7,09 11,57	24,999
2	1 2 3	3 2 9	11 16 9	0.4 0.7 0.7	5 4 3	65	4,45 4,59 8,12	64,956
3	1 2 3	3 3 4	16 7 18	0,7 0,7 0,2	3 4 1	46	6,1 3,62 13,2	45,993
4	1 2 3	9 10 3	8 19 7	0,4 0,6 0,4	5 5 5	72	4,58 7,34 2,47	71,962

SIMULACIÓN DE DATOS

Modelo clásico de cantidad económica de varios artículos con limitación de almacén

Parámetros	Articulo N°	Tasa de demanda	Costo de preparación de pedido	Costo unitario/ u. Tiempo de almacenamiento	Área de almacenamiento por unidad	Área máxima disponible de almacenamiento para los n artículos	CANTIDAD DE PEDIDO	ÁREA TOTAL OCUPADA
5	1 2 3	8 4 7	17 13 16	0,3 0,2 0,3	4 2 4	74	7,9 6,83 7,17	73,957
6	1 2 3	8 3 9	19 20 19	0,8 0,2 0,2	4 1 5	70	6,82 8,57 6,82	69,955
7	1 2 3	2 8 9	13 10 9	0,5 0,8 0,5	2 4 3	72	6,12 7,88 9,4	71,977
8	1 2 3	10 10 2	5 9 18	0,6 0,1 0,2	4 1 2	25	2,74 7,41 3,31	24,994
9	1 2 3	8 2 1	10 19 16	0,1 0,7 0,6	1 5 3	71	22,72 6,59 5,1	70,97
10	1 2 3	3 8 2	9 20 16	0,8 0,7 0,3	5 4 3	67	3,32 8,98 4,82	66,98
11	1 2 3	8 2 3	17 20 17	0,4 0,4 0,8	3 4 5	77	10,61 5,09 4,95	76,949
12	1 2 3	2 9 7	17 5 15	0,2 0,4 0,5	4 1 5	49	3,56 7,29 5,49	48,995
13	1 2 8	9 1 10	13 15 6	0,1 0,4 0,2	1 2 3	74	26,87 5,95 11,73	73,963
14	1 2 3	6 4 1	6 10 20	0,3 0,4 0,5	2 2 3	23	3,89 4,06 2,36	22,998
15	1 2 3	1 9 1	20 15 15	0,6 0,7 0,5	3 5 5	34	2,05 4,17 1,4	33,999
16	1 2 3	4 7 4	8 18 5	0,9 0,8 0,3	3 5 5	46	3,23 5,15 2,11	45,994

Modelo clásico de cantidad económica de varios artículos con limitación de almacén

Parámetros	Articulo N°	Tasa de demanda	Costo de preparación de pedido	Costo unitario/ u. Tiempo de almacenamiento	Área de almacenamiento por unidad	Área máxima disponible de almacenamiento para los n artículos	CANTIDAD DE PEDIDO	ÁREA TOTAL OCUPADA
17	1 2 3	9 5 7	7 19 9	0,4 0,8 0,1	3 5 2	69	6,72 6,3 8,63	68,95
18	1 2 3	10 9 6	17 14 16	0,6 0,5 0,4	4 3 1	65	7,78 7,69 10,77	64,966
19	1 2 3	4 5 7	18 11 18	0,8 0,5 0,2	2 3 2	60	8 6,42 12,37	59,987
20	1 2 3	4 2 1	18 7 20	0,5 0,8 0,3	4 2 4	28	3,83 2,26 2,04	27,992
21	1 2 3	7 4 1	15 7 20	0,4 0,8 0,6	3 2 1	55	12,06 6,12 6,5	54,91
22	1 2 3	8 6 8	17 6 12	0,3 0,6 0,6	3 5 2	48	6,92 2,74 6,77	47,988
23	1 2 3	1 4 9	8 14 5	0,9 0,7 0,5	3 3 5	35	1,68 4,53 3,27	34,992
24	1 2 3	6 5 8	17 12 7	0,6 0,3 0,3	4 3 2	54	6,13 5,53 6,43	53,986
25	1 2 3	4 8 7	17 10 11	0,2 0,8 0,2	2 5 2	63	8,44 5,62 8,98	62,942
26	1 2 3	1 8 5	5 7 20	0,8 0,1 0,8	2 4 5	49	1,74 4,7 5,34	48,965
27	1 2 3	7 4 9	15 12 6	0,7 0,3 0,9	3 3 5	22	2,81 1,91 1,57	21,999
28	1 2 3	3 1 7	19 19 11	0,8 0,9 0,5	2 1 5	52	7,4 4,96 6,44	51,97

Modelo clásico de cantidad económica de varios artículos con limitación de almacén

Parámetros	Articulo N°	Tasa de demanda	Costo de preparación de pedido	Costo unitario/ u. Tiempo de almacenamiento	Área de almacenamiento por unidad	Área máxima disponible de almacenamiento para los n artículos	CANTIDAD DE PEDIDO	ÁREA TOTAL OCUPADA
29	1 2 3	6 8 1	14 13 10	0,4 0,8 0,2	3 3 2	33	4,64 5,04 1,97	32,997
30	1 2 3	2 3 6	17 12 17	0,5 0,5 0,8	3 4 2	21	1,96 1,75 4,07	20,997
31	1 2 3	7 8 9	15 12 5	0,2 0,6 0,8	3 2 4	24	3,18 3,66 1,78	23,998
32	1 2 3	7 2 10	6 7 16	0,2 0,7 0,6	3 5 4	77	6,72 2,84 10,66	76,993
33	1 2 3	9 2 9	13 7 7	0,3 0,4 0,8	3 3 2	37	6,49 2,23 5,41	36,984
34	1 2 3	4 8 9	6 14 10	0,9 0,4 0,6	4 1 5	28	1,65 6,98 2,89	27,999
35	1 2 3	1 9 10	5 12 10	0,3 0,7 0,8	1 5 1	58	3,59 8,37 12,52	57,981
36	1 2 3	6 3 6	16 6 13	0,5 0,1 0,4	3 1 4	64	9,08 7,12 7,41	63,999
37	1 2 3	6 6 4	18 14 13	0,4 0,5 0,6	3 5 3	44	5,18 3,56 3,55	43,995
38	1 2 3	9 3 6	7 17 8	0,2 0,5 0,2	3 4 1	71	9,78 7,16 12,96	70,962
39	1 2 3	5 10 7	7 19 18	0,5 0,8 0,7	3 3 2	41	2,85 6,53 6,42	40,989
40	1 2 3	10 4 1	14 15 17	0,7 0,8 0,2	5 1 2	71	9,95 9,89 5,69	70,998

Modelo clásico de cantidad económica de varios artículos con limitación de almacén

Parámetros	Articulo N°	Tasa de demanda	Costo de preparación de pedido	Costo unitario/ u. Tiempo de almacenamiento	Área de almacenamiento por unidad	Área máxima disponible de almacenamiento para los n artículos	CANTIDAD DE PEDIDO	ÁREA TOTAL OCUPADA
41	1 2 3	2 5 3	14 9 9	0,5 0,7 0,9	1 4 5	65	8,82 7,55 5,2	64,989
42	1 2 3	7 2 8	15 14 8	0,1 0,7 0,6	3 2 2	45	7,68 4,32 6,64	44,968
43	1 2 3	9 1 5	17 11 19	0,8 0,2 0,2	1 2 3	63	16,28 4,92 12,27	62,939
44	1 2 3	4 9 5	8 17 13	0,1 0,2 0,8	1 2 5	73	10,23 15,81 6,22	72,95
45	1 2 3	3 2 6	6 6 6	0,5 0,5 0,2	1 3 5	42	5,98 3,44 5,13	41,937
46	1 2 3	8 9 1	15 7 6	0,2 0,4 0,9	3 2 1	47	9,52 7,87 2,64	46,954
47	1 2 3	1 6 5	9 10 14	0,3 0,3 0,5	5 4 1	31	1,4 4,03 7,85	30,996
48	1 2 3	2 10 4	12 8 19	0,2 0,3 0,7	4 3 5	67	3,78 7,74 5,72	66,929
49	1 2 3	5 6 4	20 7 9	0,8 0,7 0,5	3 3 3	32	4,72 3,07 2,88	31,996
50	1 2 3	2 9 6	19 18 20	0,4 0,3 0,1	5 5 1	47	2,43 5,04 9,63	49,991
51	1 2 3	8 7 1	7 7 7	0,4 0,6 0,3	4 3 1	43	5,57 5,71 3,56	42,98
52	1 2 3	9 8 4	7 19 19	0,5 0,9 0,4	3 5 1	69	6,45 7,71 11,06	68,986

Modelo clásico de cantidad económica de varios artículos con limitación de almacén

Parámetros	Articulo N°	Tasa de demanda	Costo de preparación de pedido	Costo unitario/ u. Tiempo de almacenamiento	Área de almacenamiento por unidad	Área máxima disponible de almacenamiento para los n artículos	CANTIDAD DE PEDIDO	ÁREA TOTAL OCUPADA
53	1 2 3	1 1 9	7 14 7	0,2 0,7 0,3	4 3 5	25	1,13 1,79 3,02	24,999
54	1 2 3	3 2 5	15 10 16	0,2 0,7 0,9	1 4 4	28	5,61 1,88 3,72	27,997
55	1 2 3	9 8 9	9 8 17	0,2 0,1 0,2	5 1 5	23	1,66 3,29 2,28	22,998
56	1 2 3	1 9 9	12 11 5	0,5 0,3 0,9	3 1 5	51	3,4 15,5 5,06	50,994
57	1 2 3	3 10 10	15 19 9	0,4 0,4 0,3	2 2 3	55	5,67 11,64 6,79	54,979
58	1 2 3	10 8 5	18 13 13	0,3 0,7 0,4	4 2 1	21	2,87 3,05 3,4	20,999
59	1 2 3	2 7 4	14 14 9	0,6 0,7 0,8	3 4 4	41	3,05 4,97 2,99	40,986
60	1 2 3	9 5 1	19 16 10	0,8 0,8 0,2	3 5 4	62	9,28 5,13 2,12	61,956
61	1 2 3	6 3 4	19 18 18	0,6 0,8 0,5	2 2 1	32	7,23 4,87 7,78	31,989
62	1 2 3	7 3 10	10 9 13	0,9 0,5 0,3	3 2 4	70	7,02 5,48 9,48	69,942
63	1 2 3	8 4 6	16 14 19	0,5 0,5 0,7	4 4 1	42	5,03 3,33 8,57	41,986
64	1 2 3	7 7 8	17 20 6	0,6 0,9 0,4	2 5 3	69	9,49 6,83 5,27	68,948

Modelo clásico de cantidad económica de varios artículos con limitación de almacén

Parámetros	Articulo N°	Tasa de demanda	Costo de preparación de pedido	Costo unitario/ u. Tiempo de almacenamiento	Área de almacenamiento por unidad	Área máxima disponible de almacenamiento para los n artículos	CANTIDAD DE PEDIDO	ÁREA TOTAL OCUPADA
65	1 2 3	9 5 2	10 8 6	0,3 0,3 0,7	2 2 2	48	12,09 8,06 3,83	47,961
66	1 2 3	2 9 2	7 20 8	0,6 0,2 0,7	2 3 4	25	1,77 5,34 1,36	24,997
67	1 2 3	5 4 9	9 20 7	0,5 0,1 0,3	4 2 1	40	3,96 7,66 8,84	39,997
68	1 2 3	8 9 2	14 19 19	0,9 0,5 0,9	3 4 4	56	5,88 6,57 3,02	55,992
69	1 2 3	5 2 2	5 17 12	0,2 0,4 0,3	2 1 1	37	8,6 10,32 9,45	36,97
70	1 2 3	9 1 2	9 19 9	0,5 0,1 0,7	4 5 3	51	6,5 2,98 3,35	50,969
71	1 2 3	3 10 1	17 8 7	0,2 0,3 0,6	4 5 5	51	4,56 5,08 1,47	50,992
72	1 2 3	7 2 6	14 16 9	0,8 0,4 0,6	2 2 5	42	7,29 4,41 3,72	41,985
73	1 2 3	8 2 3	6 15 5	0,8 0,6 0,2	4 5 2	47	5,02 3,71 4,19	46,972
74	1 2 3	5 9 3	5 14 18	0,6 0,6 0,3	1 4 3	46	5,07 6,49 4,99	45,995
75	1 2 3	8 7 1	17 6 19	0,5 0,1 0,8	4 2 3	67	9,78 8,13 3,85	66,934
76	1 2 3	5 9 4	17 7 18	0,2 0,3 0,3	1 3 3	63	14,63 7,78 8,32	62,92

Modelo clásico de cantidad económica de varios artículos con limitación de almacén

Parámetros	Articulo N°	Tasa de demanda	Costo de preparación de pedido	Costo unitario/ u. Tiempo de almacenamiento	Área de almacenamiento por unidad	Área máxima disponible de almacenamiento para los n artículos	CANTIDAD DE PEDIDO	ÁREA TOTAL OCUPADA
77	1 2 3	4 6 8	11 9 6	0,6 0,3 0,2	2 5 5	66	6,68 5,38 5,14	65,978
78	1 2 3	10 2 9	18 17 15	0,6 0,3 0,6	4 4 3	50	5,72 2,52 5,67	49,993
79	1 2 3	2 5 9	17 10 9	0,1 0,4 0,8	5 3 1	53	4,54 6,57 10,61	53
80	1 2 3	10 9 5	16 14 13	0,1 0,3 0,2	5 2 3	54	5,31 7,24 4,32	53,992
81	1 2 3	7 10 4	6 19 17	0,6 0,4 0,5	5 3 3	73	4,05 11,06 6,51	72,994
82	1 2 3	6 2 4	7 9 10	0,3 0,7 0,8	5 5 3	31	2,62 1,68 3,16	30,996
83	1 2 3	3 8 2	9 16 8	0,8 0,7 0,4	1 4 4	51	5,87 8,24 3,04	50,965
84	1 2 3	10 8 2	5 8 8	0,3 0,5 0,8	5 2 4	74	7,35 10,69 3,96	73,984
85	1 2 3	6 10 5	5 19 19	0,8 0,6 0,8	5 3 1	35	1,91 6,17 6,95	34,995
86	1 2 3	8 3 3	10 10 5	0,9 0,1 0,5	2 5 2	78	11,59 8,5 6,15	77,992
87	1 2 3	6 3 1	19 18 18	0,9 0,8 0,9	1 5 5	66	13,65 6,68 3,78	65,949
88	1 2 3	2 7 2	20 8 20	0,2 0,2 0,2	5 3 3	23	1,71 2,61 2,2	22,999

Modelo clásico de cantidad económica de varios artículos con limitación de almacén

Parámetros	Articulo N°	Tasa de demanda	Costo de preparación de pedido	Costo unitario/ u. Tiempo de almacenamiento	Área de almacenamiento por unidad	Área máxima disponible de almacenamiento para los n artículos	CANTIDAD DE PEDIDO	ÁREA TOTAL OCUPADA
89	1 2 3	2 4 1	10 10 5	0,5 0,5 0,4	1 1 2	45	8,94 12,65 5	31,593
90	1 2 3	10 9 3	20 19 18	0,4 0,4 0,1	1 4 1	35	10,1 4,86 5,46	34,994
91	1 2 3	5 7 6	16 19 10	0,6 0,2 0,3	3 5 2	54	5,21 5,43 5,6	53,983
92	1 2 3	4 10 10	17 14 6	0,7 0,5 0,5	3 2 5	52	5,06 8,84 3,83	51,983
93	1 2 3	9 2 6	5 7 5	0,6 0,4 0,4	5 2 4	34	3,27 2,82 3	33,991
94	1 2 3	10 3 9	20 17 14	0,2 0,5 0,5	1 3 2	28	9,35 2,74 5,22	27,995
95	1 2 3	5 4 9	20 6 10	0,3 0,6 0,8	2 4 5	24	3,76 1,3 2,25	23,998
96	1 2 3	4 9 4	15 18 16	0,3 0,5 0,5	2 4 2	50	5,75 6,73 5,78	49,983
97	1 2 3	6 4 7	7 14 15	0,7 0,7 0,3	2 2 3	25	3,01 3,48 4	24,995
98	1 2 3	9 8 4	5 18 18	0,1 0,5 0,4	4 2 5	43	2,98 7,22 3,33	42,987
99	1 2 3	8 2 9	17 12 8	0,8 0,7 0,1	4 2 4	26	3,2 1,88 2,36	26
100	1 2 3	5 10 4	16 6 8	0,6 0,6 0,8	1 5 3	40	9,03 4,02 3,62	39,984

Modelo sin costo de preparación

SIMULACIÓN DE DATOS

Modelo dinámico de cantidad económica de pedido sin C. de preparación

							CANTIDAD DE PEDIDO			C. MINIMO DE INVENTARIO
Parámetros	Mes	Demanda	Capacidad	Capacidad extra	C. producción tiempo normal	C. producción tiempo extra	C. de almacenamiento	Periodo normal	Periodo extra	
1	1	90	50	98	5,11	8,64	0,1	90	8	4191,36
	2	100	60	185				100	85	
	3	120	80	208				120	88	
	4	110	70	166				110	56	
2	1	205	101	70	6,86	9,03	0,14	101	104	**4973,13**
	2	203	114	65				114	89	
	3	119	95	47				95	24	
	4	124	108	63				108	16	
3	1	125	94	47	6,42	8,86	0,14	94	31	**3386,78**
	2	122	118	50				118	4	
	3	105	85	64				85	20	
	4	136	87	74				87	49	
4	1	177	100	64	6,47	9,94	0,09	100	77	**3759,72**
	2	103	89	56				89	14	
	3	117	128	70				117	0	
	4	133	129	71				129	4	
5	1	124	124	75	5,54	8,89	0,07	124	0	**3778,2**
	2	171	123	81				123	48	
	3	91	84	46				84	7	
	4	194	80	48				80	114	
6	1	105	108	49	6,35	8,25	0,12	108	0	**3962,46**
	2	137	109	67				109	25	
	3	153	120	79				120	33	
	4	183	88	78				88	95	
7	1	105	108	49	6,35	8,25	0,12	108	0	**3962,46**
	2	137	109	67				109	25	
	3	153	120	79				120	33	
	4	183	88	78				88	95	
8	1	161	105	69	5,21	9,26	0,15	105	56	**3775,95**
	2	161	128	63				128	33	

Modelo dinámico de cantidad económica de pedido sin C. de preparación

							CANTIDAD DE PEDIDO			C. MINIMO DE INVENTARIO
Parámetros	Mes	Demanda	Capacidad	Capacidad extra	C. producción tiempo normal	C. producción tiempo extra	C. de almacenamiento	Periodo normal	Periodo extra	
	3	163	124	73				124	39	
	4	133	124	42				124	9	
9	1	103	98	76	5,99	9,15	0,1	98	5	**3232,04**
	2	129	123	74				123	6	
	3	117	88	85				88	29	
	4	146	102	60				102	44	
10	1	204	123	88	6,17	9,23	0,09	123	81	**4904,16**
	2	170	102	48				102	68	
	3	192	84	54				84	108	
	4	101	113	74				101	0	
11	1	188	80	76	6,24	8,95	0,15	80	108	**4784,26**
	2	141	124	45				124	17	
	3	130	103	85				103	27	
	4	198	97	60				97	101	
12	1	102	127	44	5,52	9,25	0,12	127	0	**2794,08**
	2	158	89	75				89	44	
	3	112	97	86				97	15	
	4	94	112	82				94	0	
13	1	154	103	71	6,26	8,19	0,1	103	51	**3490,53**
	2	90	115	45				90	0	
	3	148	117	76				117	31	
	4	127	84	62				84	43	
14	1	208	123	76	6,21	8,51	0,1	123	85	**5130,6**
	2	151	89	61				89	62	
	3	186	122	59				122	64	
	4	183	129	57				129	54	
15	1	167	112	42	5,03	8,75	0,1	112	55	**3968,07**
	2	155	87	53				87	68	
	3	207	109	82				109	98	
	4	96	107	76				96	0	
16	1	202	127	53	5,37	9,98	0,15	127	75	**4736,93**
	2	103	107	81				103	0	
	3	202	108	79				108	94	
	4	161	81	52				81	80	
17	1	187	92	84	6,83	9,94	0,06	92	95	**4266,63**

Modelo dinámico de cantidad económica de pedido sin C. de preparación

							CANTIDAD DE PEDIDO		C. MINIMO DE INVENTARIO
Parámetros	Mes	Demanda	Capacidad	Capacidad extra	C. producción tiempo normal	C. producción tiempo extra	C. de almacenamiento	Periodo normal	Periodo extra	
	2	97	105	56				97	0	
	3	133	127	68				127	6	
	4	136	80	51				80	56	
18	1	163	97	52	5,32	8,62	0,15	97	66	**3326,12**
	2	121	97	61				97	24	
	3	147	85	44				85	62	
	4	100	109	72				100	0	
19	1	168	114	58	5,98	9,82	0,14	114	54	**4534,25**
	2	173	84	87				84	89	
	3	129	106	67				106	23	
	4	146	91	74				91	55	
20	1	191	102	68	6,42	8,14	0,06	102	89	**4101,68**
	2	135	106	47				106	29	
	3	114	120	71				114	0	
	4	155	110	40				110	45	
21	1	148	119	45	6,7	9,56	0,06	119	29	**4592,18**
	2	178	90	70				90	88	
	3	121	91	55				91	30	
	4	147	80	44				80	67	
22	1	183	126	58	6,66	9,77	0,12	126	57	**4910,38**
	2	134	83	42				83	51	
	3	208	85	74				85	123	
	4	100	90	78				90	10	
23	1	101	127	88	5,94	9,89	0,06	127	0	**3637,25**
	2	155	116	74				116	13	
	3	139	109	40				109	30	
	4	155	105	58				105	50	
24	1	90	50	98	5,11	8,64	0,1	90	8	9320,67
	2	100	60	185				100	85	
	3	120	80	208				120	88	
	4	110	70	166				110	56	
25	1	160	117	70	6,66	8,19	0,12	117	43	**5244,42**
	2	155	99	50				99	56	
	3	190	95	51				95	95	
	4	96	80	57				80	16	

Modelo dinámico de cantidad económica de pedido sin C. de preparación

							CANTIDAD DE PEDIDO		C. MINIMO DE INVENTARIO
Parámetros	Mes	Demanda	Capacidad	Capacidad extra	C. producción tiempo normal	C. producción tiempo extra	C. de almacenamiento	Periodo normal	Periodo extra	
26	1	144	89	79	6,65	9,71	0,08	89	25	**4765,67**
	2	158	97	53				97	61	
	3	154	129	62				129	25	
	4	190	82	60				82	108	
27	1	181	121	87	6,02	9,42	0,09	121	60	**4551,64**
	2	93	103	42				93	0	
	3	195	123	52				123	72	
	4	177	114	60				114	63	
28	1	133	118	42	5,45	9,6	0,1	118	15	**3637,22**
	2	174	129	58				129	45	
	3	126	91	88				91	35	
	4	144	120	74				120	24	
29	1	144	111	71	6,45	8	0,1	111	33	**4163,05**
	2	127	110	55				110	17	
	3	199	94	56				94	105	
	4	136	127	40				127	9	
30	1	118	124	74	6,39	8,42	0,08	124	0	**3849,03**
	2	94	80	88				80	8	
	3	203	127	72				127	76	
	4	153	129	69				129	24	
31	1	173	111	58	6,01	9,71	0,12	111	62	**4456,7**
	2	158	125	78				125	33	
	3	107	96	59				96	11	
	4	184	96	53				96	88	
32	1	98	98	56	5,07	9,89	0,13	98	0	**2819,68**
	2	121	81	84				81	40	
	3	134	81	45				81	53	
	4	101	87	85				87	14	
33	1	129	114	49	6,65	9,57	0,05	114	15	**3611,14**
	2	118	124	53				118	0	
	3	113	129	71				113	0	
	4	148	83	79				83	65	
34	1	109	91	53	5,68	9,33	0,08	91	18	**4862,92**
	2	169	82	55				82	87	
	3	185	99	58				99	86	

Modelo dinámico de cantidad económica de pedido sin C. de preparación

							CANTIDAD DE PEDIDO		C. MINIMO DE INVENTARIO
Parámetros	Mes	Demanda	Capacidad	Capacidad extra	C. producción tiempo normal	C. producción tiempo extra	C. de almacenamiento	Periodo normal	Periodo extra	
	4	209	114	77				114	95	
35	1	191	99	66	6,99	9,81	0,14	99	92	**5030,99**
	2	155	109	85				109	46	
	3	108	101	56				101	7	
	4	179	109	72				109	70	
36	1	174	105	61	6,08	9,68	0,07	105	69	**4140,65**
	2	125	102	77				102	23	
	3	163	114	74				114	49	
	4	131	124	43				124	7	
37	1	170	124	59	6,32	8,94	0,08	124	46	**4088,47**
	2	137	80	68				80	57	
	3	156	81	58				81	75	
	4	110	122	66				110	0	
38	1	190	96	45	6,74	8,02	0,15	96	94	**4126,83**
	2	140	111	87				111	29	
	3	127	129	89				127	0	
	4	124	83	51				83	41	
39	1	173	119	69	6,85	9,55	0,08	119	54	**5140,07**
	2	122	92	86				92	30	
	3	180	83	66				83	97	
	4	173	94	45				94	79	
40	1	138	120	40	5,09	9,24	0,07	120	18	**2998,26**
	2	160	97	67				97	63	
	3	121	128	69				121	0	
	4	99	93	56				93	6	
41	1	203	129	57	5,75	8,1	0,15	129	74	**4977,45**
	2	186	81	60				81	105	
	3	153	118	43				118	35	
	4	200	112	87				112	88	
42	1	125	95	58	5,78	9,92	0,15	95	30	**3559,59**
	2	178	101	49				101	77	
	3	94	120	82				94	0	
	4	136	127	80				127	9	
43	1	100	109	72	6,06	8,85	0,06	109	0	**4286,1**
	2	188	107	83				107	72	

Modelo dinámico de cantidad económica de pedido sin C. de preparación

							CANTIDAD DE PEDIDO			C. MINIMO DE INVENTARIO
Parámetros	Mes	Demanda	Capacidad	Capacidad extra	C. producción tiempo normal	C. producción tiempo extra	C. de almacenamiento	Periodo normal	Periodo extra	
	3	152	121	44				121	31	
	4	177	84	47				84	93	
44	1	184	102	42	6,91	8,22	0,12	102	82	**4201,75**
	2	113	110	67				110	3	
	3	186	88	56				88	98	
	4	90	88	41				88	2	
45	1	94	93	66	5,02	9,96	0,14	93	1	**4118,88**
	2	151	98	73				98	53	
	3	206	98	79				98	108	
	4	154	97	70				97	57	
46	1	129	100	53	6,35	8,97	0,1	100	29	**4129,05**
	2	154	109	48				109	45	
	3	117	127	78				117	0	
	4	191	121	42				121	70	
47	1	111	112	42	6,63	9,84	0,15	112	0	**3557,57**
	2	119	123	53				118	0	
	3	109	119	41				109	0	
	4	174	126	72				126	48	
48	1	177	94	66	6,59	9,89	0,08	94	83	**4632,91**
	2	92	113	46				92	0	
	3	130	125	75				125	5	
	4	202	86	73				86	116	
49	1	141	92	89	6,58	8,54	0,12	92	49	**4437,23**
	2	155	110	78				110	45	
	3	145	115	61				115	30	
	4	171	85	49				85	86	
50	1	194	105	45	5,55	9,81	0,13	105	89	**5457,96**
	2	198	115	84				115	83	
	3	161	81	40				81	80	
	4	170	83	55				83	87	
51	1	159	123	77	5,51	8,24	0,14	123	36	**3769,17**
	2	94	84	89				84	10	
	3	175	126	53				126	49	
	4	180	122	43				122	58	
52	1	102	89	83	5,33	9,07	0,12	89	13	**3679,49**

Modelo dinámico de cantidad económica de pedido sin C. de preparación

							CANTIDAD DE PEDIDO		C. MINIMO DE INVENTARIO
Parámetros	Mes	Demanda	Capacidad	Capacidad extra	C. producción tiempo normal	C. producción tiempo extra	C. de almacenamiento	Periodo normal	Periodo extra	
	2	157	103	74				103	54	
	3	156	93	70				93	63	
	4	154	111	68				111	43	
53	1	113	104	44	6,27	8,56	0,08	104	9	**4586,04**
	2	196	92	61				92	104	
	3	151	119	70				119	32	
	4	184	90	75				90	94	
54	1	160	91	54	6,92	9,25	0,11	91	69	**5761,33**
	2	204	82	62				82	122	
	3	201	103	46				103	98	
	4	154	107	59				107	47	
55	1	202	88	62	5,5	9,69	0,1	88	114	**5225,2**
	2	201	127	89				127	74	
	3	132	118	73				118	14	
	4	203	127	58				127	76	
56	1	165	115	65	5,04	9,35	0,14	115	50	**3884,31**
	2	121	96	76				96	25	
	3	193	123	44				123	70	
	4	139	105	51				105	34	
57	1	191	96	47	6,36	9,23	0,13	96	95	**4590,63**
	2	120	89	72				89	31	
	3	177	105	73				105	72	
	4	128	92	70				92	36	
58	1	161	129	52	6,52	8,89	0,11	129	32	**4590,63**
	2	92	115	53				92	0	
	3	134	95	45				95	39	
	4	117	105	85				105	12	
59	1	109	91	77	5,33	9,12	0,06	91	18	**3481,07**
	2	105	91	82				91	14	
	3	165	96	80				96	69	
	4	109	125	71				109	0	
60	1	185	91	68	5,65	9,34	0,05	91	94	**2981,84**
	2	143	85	82				85	58	
	3	191	97	55				97	94	
	4	156	96	82				96	60	

Modelo dinámico de cantidad económica de pedido sin C. de preparación

							CANTIDAD DE PEDIDO		C. MINIMO DE INVENTARIO
Parámetros	Mes	Demanda	Capacidad	Capacidad extra	C. producción tiempo normal	C. producción tiempo extra	C. de almacenamiento	Periodo normal	Periodo extra	
61	1	156	83	68	6,42	8,63	0,14	83	73	**4942,55**
	2	182	96	76				96	86	
	3	96	94	72				94	2	
	4	96	91	87				91	5	
62	1	198	122	82	6,24	8,73	0,1	122	76	**3768,74**
	2	113	86	81				86	27	
	3	143	97	67				97	46	
	4	142	104	43				104	38	
63	1	144	87	55	6,31	9,83	0,14	87	57	**4184,85**
	2	134	126	49				126	8	
	3	167	106	68				106	61	
	4	155	97	88				97	58	
64	1	137	108	42	5,41	9,69	0,11	108	29	**4432,17**
	2	146	122	49				112	34	
	3	152	123	54				123	29	
	4	131	84	61				84	47	
65	1	130	126	42	5,68	8,91	0,13	126	4	**3658,25**
	2	162	82	56				82	80	
	3	170	99	69				99	71	
	4	178	81	59				81	97	
66	1	150	83	88	6,89	9,27	0,07	83	67	**4449,29**
	2	188	93	84				93	95	
	3	105	127	67				105	0	
	4	198	97	79				97	101	
67	1	183	88	73	5,93	9,1	0,06	88	95	**5039,53**
	2	190	93	68				93	97	
	3	204	82	79				82	122	
	4	144	109	80				109	35	
68	1	142	129	45	5,51	9,09	0,08	129	13	**5384,59**
	2	164	124	86				124	40	
	3	202	92	47				92	110	
	4	171	80	61				80	91	
69	1	95	81	51	6,89	9,4	0,09	81	14	**4650,22**
	2	184	81	50				81	103	
	3	190	127	56				127	63	

Modelo dinámico de cantidad económica de pedido sin C. de preparación

							CANTIDAD DE PEDIDO		C. MINIMO DE INVENTARIO
Parámetros	Mes	Demanda	Capacidad	Capacidad extra	C. producción tiempo normal	C. producción tiempo extra	C. de almacenamiento	Periodo normal	Periodo extra	
	4	186	95	71				95	91	
70	1	137	103	51	5,16	8,23	0,07	103	34	**5191,36**
	2	119	105	62				105	14	
	3	114	105	61				105	9	
	4	105	128	53				105	0	
71	1	112	123	64	5,6	9,23	0,08	123	0	**2627,01**
	2	170	105	44				105	54	
	3	185	126	63				126	59	
	4	174	90	72				90	84	
72	1	205	119	72	5,6	9,72	0,05	119	86	**4306,68**
	2	97	118	82				97	0	
	3	102	80	43				80	22	
	4	125	121	86				121	4	
73	1	165	87	89	6,58	9,08	0,14	87	78	**3424,61**
	2	158	106	56				106	52	
	3	117	91	51				91	26	
	4	117	88	55				88	29	
74	1	116	95	59	6,48	8,78	0,08	95	21	**4129,08**
	2	150	90	62				90	60	
	3	173	101	62				101	72	
	4	174	86	78				86	88	
75	1	101	98	40	6,18	9,75	0,1	98	3	**4525,85**
	2	109	127	42				109	0	
	3	197	112	67				112	85	
	4	108	88	51				88	20	
76	1	147	120	57	6,8	9,86	0,06	120	27	**3568,6**
	2	144	122	65				122	22	
	3	135	99	49				99	36	
	4	122	99	73				99	23	
77	1	203	115	83	5,65	8,01	0,14	115	88	**4056,17**
	2	112	118	60				112	0	
	3	101	95	70				95	6	
	4	146	80	54				80	66	
78	1	178	121	78	5,73	9,22	0,11	121	57	**3550,88**
	2	154	81	70				81	73	

Modelo dinámico de cantidad económica de pedido sin C. de preparación

							CANTIDAD DE PEDIDO			C. MINIMO DE INVENTARIO
Parámetros	Mes	Demanda	Capacidad	Capacidad extra	C. producción tiempo normal	C. producción tiempo extra	C. de almacenamiento	Periodo normal	Periodo extra	
	3	182	113	49				113	69	
	4	126	115	81				115	11	
79	1	166	121	77	5,26	9,85	0,12	121	45	**4401,34**
	2	131	108	48				108	23	
	3	93	97	79				93	0	
	4	105	99	54				99	6	
80	1	162	84	61	5,36	8,06	0,13	84	78	**2942,43**
	2	191	122	61				122	69	
	3	113	95	72				95	18	
	4	121	84	45				84	37	
81	1	204	87	56	5,07	8,55	0,1	87	117	**3691,95**
	2	150	80	61				80	70	
	3	112	129	78				112	0	
	4	165	87	41				87	78	
82	1	92	107	82	6,41	9,93	0,12	107	0	**4121,36**
	2	116	91	62				91	10	
	3	129	99	72				99	30	
	4	162	90	57				90	72	
83	1	135	90	51	6,2	8,62	0,09	90	45	**3594,66**
	2	121	119	43				119	2	
	3	153	102	85				102	51	
	4	128	103	64				103	25	
84	1	120	124	86	6,4	8,94	0,12	124	0	**3628,38**
	2	201	93	83				93	104	
	3	190	111	57				111	79	
	4	114	127	58				114	0	
85	1	112	81	78	5,74	9,47	0,13	81	31	**4465,37**
	2	173	118	49				118	55	
	3	90	97	60				90	0	
	4	208	85	49				85	123	
86	1	159	90	84	5,46	8,66	0,06	90	69	**4124,59**
	2	110	83	46				83	27	
	3	103	125	63				103	0	
	4	135	106	42				106	29	
87	1	122	122	43	5,46	8,62	0,11	122	0	**3169,42**

Modelo dinámico de cantidad económica de pedido sin C. de preparación

							CANTIDAD DE PEDIDO		C. MINIMO DE INVENTARIO
Parámetros	Mes	Demanda	Capacidad	Capacidad extra	C. producción tiempo normal	C. producción tiempo extra	C. de almacenamiento	Periodo normal	Periodo extra	
	2	183	88	62				88	95	
	3	187	88	78				88	99	
	4	120	107	65				107	13	
88	1	140	113	72	12,72	24,79	0,27	113	27	**3995,75**
	2	157	86	67				86	71	
	3	90	89	80				89	1	
	4	179	116	74				116	63	
89	1	185	84	44	10,68	21,76	0,72	84	101	**9156,15**
	2	200	122	63				122	78	
	3	172	93	72				93	79	
	4	187	106	87				106	81	
90	1	190	85	47	13,52	17,01	0,19	85	105	**11704,46**
	2	122	98	45				98	24	
	3	130	117	49				117	13	
	4	111	118	79				111	0	
91	1	142	101	85	12,16	24,08	0,49	101	41	**7972,78**
	2	147	128	68				128	19	
	3	130	80	73				80	50	
	4	177	124	79				124	53	
92	1	100	116	73	12,39	23,17	0,15	116	0	**9189,87**
	2	128	120	66				112	0	
	3	162	108	87				108	54	
	4	194	122	78				122	72	
93	1	135	128	62	9,51	22,57	0,18	128	7	**8598,04**
	2	185	82	88				82	103	
	3	107	115	78				107	0	
	4	90	117	58				90	0	
94	1	96	111	87	14,16	24,05	0,3	111	0	**6335,35**
	2	184	112	63				112	57	
	3	201	120	52				120	81	
	4	135	87	71				87	48	
95	1	130	94	46	13,3	21,38	0,2	94	36	**10565,15**
	2	113	89	66				89	24	
	3	152	123	49				123	29	
	4	205	94	55				94	111	

Modelo dinámico de cantidad económica de pedido sin C. de preparación

							CANTIDAD DE PEDIDO		C. MINIMO DE INVENTARIO
Parámetros	Mes	Demanda	Capacidad	Capacidad extra	C. producción tiempo normal	C. producción tiempo extra	C. de almacenamiento	Periodo normal	Periodo extra	
96	1	95	84	86	11,79	24,86	0,64	84	11	**9596,79**
	2	189	116	76				116	73	
	3	90	118	42				90	0	
	4	182	129	81				129	53	
97	1	195	82	75	12,79	18,9	0,63	82	113	**8345,23**
	2	175	112	65				112	63	
	3	209	128	49				128	81	
	4	194	106	71				106	88	
98	1	180	83	73	13,24	23,13	0,18	83	97	**11994,68**
	2	138	94	59				94	44	
	3	115	82	56				82	33	
	4	130	120	87				120	10	
99	1	119	128	79	13,04	24,69	0,46	128	0	**9273,74**
	2	150	86	68				86	55	
	3	188	117	68				117	71	
	4	127	107	75				107	20	
100	1	90	50	98	5,11	8,64	0,1	90	8	9320,67
	2	100	60	185				100	85	
	3	120	80	208				120	88	
	4	110	70	166				110	56	

Modelo con preparación

SIMULACIÓN DE DATOS

Modelo dinámico de cantidad económica de pedido con costo de preparación

		Periodo	Tasa de demanda	Costo de preparación de pedido	Costo unitario/ u. Tiempo de almacenamiento	Inventario inicial	Costo de pro primeras unidades	Costo de pro unidades siguientes	Unidades de primer costo de pro	CANTIDAD DE PEDIDO	COSTO MINIMO INVENTARIO
1	**Parámetros**	1 2 3	6 4 7	12 14 12	1,33 0,19 1,42	6	18	25	9	0 11 0	124,92
2	**Parámetros**	1 2 3	5 6 6	7 6 9	1,17 0,93 2,22	4	12	27	8	9 0 4	146,56
3	**Parámetros**	1 2 3	5 3 8	10 13 14	0,93 0,56 1,34	2	15	32	3	3 3 8	335,86
4	**Parámetros**	1 2 3	5 4 6	9 11 4	2,77 1,31 1,94	5	10	28	4	2 4 4	144,35
5	**Parámetros**	1 2 3	5 8 8	14 9 8	0,83 0,52 0,31	2	15	26	2	15 2 2	479,76
6	**Parámetros**	1 2 3	2 6 5	8 10 7	0,44 2,7 1,45	2	16	25	1	0 10 1	305,68
7	**Parámetros**	1 2 3	6 7 5	8 2 2	0,18 1,75 0,83	8	11	25	8	0 9 1	84,26
8	**Parámetros**	1 2 3	6 9 7	10 10 9	1,54 1,76 0,36	2	18	33	4	5 8 7	450,84
9	**Parámetros**	1 2 3	1 5 3	10 5 10	1,8 2,23 0,79	3	13	34	4	0 3 3	102,86
10	**Parámetros**	1 2 3	7 8 7	3 14 5	1,13 1,71 2,2	1	15	28	8	9 9 3	224,06
11	**Parámetros**	1 2 3	3 4 9	14 13 2	2,46 0,92 1,05	3	15	30	9	0 10 3	148,68
12	**Parámetros**	1 2 3	7 8 6	9 14 3	1,08 1,48 2,73	1	18	31	7	8 8 4	267,5
13	**Parámetros**	1 2 3	9 4 2	7 11 3	2,9 2,15 1,54	2	17	32	9	10 1 2	165,25
14	**Parámetros**	1 2 3	9 6 6	1 9 11	0,53 0,24 2,86	4	11	28	1	15 1 1	492,82
15	**Parámetros**	1 2 3	9 4 6	5 6 10	1,82 2,6 0,35	7	17	30	4	2 5 5	222,09
16	**Parámetros**					6	17	32	6		174,83

Modelo dinámico de cantidad económica de pedido con costo de preparación

		Periodo	Tasa de demanda	Costo de preparación de pedido	Costo unitario/ u. Tiempo de almacenamiento	Inventario inicial	Costo de pro primeras unidades	Costo de pro unidades siguientes	Unidades de primer costo de pro	CANTIDAD DE PEDIDO	COSTO MINIMO INVENTARIO
		1 2 3	7 5 5	7 11 9	0,78 0,51 2,09					7 0 4	
17	**Parámetros**	1 2 3	6 3 9	5 5 3	0,38 2,86 0,27	6	17	27	9	0 10 2	117,17
18	**Parámetros**	1 2 3	8 8 4	5 3 8	1,7 1,12 0,72	9	12	28	6	0 7 4	143,42
19	**Parámetros**	1 2 3	3 5 3	3 12 7	2,1 2,79 1,73	6	18	31	7	0 8 0	123,35
20	**Parámetros**	1 2 3	3 3 3	1 13 5	2,81 0,79 1,34	2	17	32	7	8 0 0	98,51
21	**Parámetros**	1 2 3	6 1 2	10 8 5	1,05 2,29 2,14	4	16	29	9	3 0 2	101,49
22	**Parámetros**	1 2 3	4 5 3	12 3 6	0,88 2,53 0,64	9	11	32	8	0 3 0	58,49
23	**Parámetros**	1 2 3	6 2 1	12 1 10	2,54 2,58 2,63	8	11	29	5	0 1 0	40,11
24	**Parámetros**	1 2 3	4 2 1	4 12 1	0,87 0,57 1,45	4	12	32	8	2 0 1	45,62
25	**Parámetros**	1 2 3	9 7 4	5 9 4	1,04 1,56 0,58	4	13	30	5	6 6 4	221,72
26	**Parámetros**	1 2 3	9 9 1	3 2 12	1,71 2,97 2,36	6	12	29	9	3 10 0	123,62
27	**Parámetros**	1 2 3	7 2 9	14 2 11	1,8 1,66 2,29	7	14	30	1	1 1 9	353,26
28	**Parámetros**	1 2 3	2 1 5	5 8 2	1,74 2,15 0,43	1	15	33	8	2 0 5	115,89
29	**Parámetros**	1 2 3	6 7 5	13 3 3	1,19 1,11 1,24	1	16	27	1	15 1 1	489,25
30	**Parámetros**	1 2 3	1 7 5	13 7 2	2,75 2,66 1,7	8	18	27	7	0 0 8	114,62
31	**Parámetros**	1 2 3	5 1 2	7 9 11	0,76 2,22 0,66	6	19	30	1	1 1 0	64,32

Modelo dinámico de cantidad económica de pedido con costo de preparación

		Periodo	Tasa de demanda	Costo de preparación de pedido	Costo unitario/ u. Tiempo de almacenamiento	Inventario inicial	Costo de pro primeras unidades	Costo de pro unidades siguientes	Unidades de primer costo de pro	CANTIDAD DE PEDIDO	COSTO MINIMO INVENTARIO
32	**Parámetros**	1 2 3	1 7 5	13 7 2	2,75 2,66 1,7	8	18	27	7	0 0 8	114,62
33	**Parámetros**	1 2 3	5 1 2	7 9 11	0,76 2,22 0,66	6	19	30	1	1 1 0	64,32
34	**Parámetros**	1 2 3	1 5 6	13 3 4	0,74 2,72 1,33	8	19	32	4	0 0 4	107,62
35	**Parámetros**	1 2 3	2 8 9	14 10 1	0,27 1,22 0,51	4	12	34	7	0 8 8	175,54
36	**Parámetros**	1 2 3	8 9 7	14 7 5	1,94 1,48 2,89	6	14	33	9	10 1 10	223,48
37	**Parámetros**	1 2 3	4 7 1	8 7 3	1,77 2,04 2,12	8	10	27	5	0 4 0	71,44
38	**Parámetros**	1 2 3	1 3 1	10 6 3	2,37 0,24 0,64	2	16	33	7	0 3 0	59,62
39	**Parámetros**	1 2 3	2 6 8	1 4 1	1,04 1,13 2,36	8	12	32	1	1 1 6	254,95
40	**Parámetros**	1 2 3	1 1 8	5 14 6	0,18 2,47 0,7	9	14	32	3	0 0 1	46,28
41	**Parámetros**	1 2 3	8 4 4	11 8 1	0,12 2,72 1,5	7	12	29	5	1 4 4	128,84
42	**Parámetros**	1 2 3	5 1 8	12 13 4	0,11 2,85 0,89	3	12	30	2	2 2 7	270,22
43	**Parámetros**	1 2 3	6 2 6	6 13 3	1,76 2,37 2,66	8	17	30	7	0 0 8	98,82
44	**Parámetros**	1 2 3	7 8 9	10 4 1	2,6 2,53 1,36	2	18	30	8	9 4 9	258,32
45	**Parámetros**	1 2 3	2 9 6	2 8 4	2,02 2,52 0,63	2	13	29	8	0 9 9	158,04
46	**Parámetros**	1 2 3	2 1 5	8 11 9	1,34 1,6 2,61	3	11	29	7	0 0 5	69,62
47	**Parámetros**					2	13	32	7		266,14

Modelo dinámico de cantidad económica de pedido con costo de preparación

		Periodo	Tasa de demanda	Costo de preparación de pedido	Costo unitario/ u. Tiempo de almacenamiento	Inventario inicial	Costo de pro primeras unidades	Costo de pro unidades siguientes	Unidades de primer costo de pro	CANTIDAD DE PEDIDO	COSTO MINIMO INVENTARIO
		1 2 3	8 9 9	9 4 11	3 2,18 0,78					8 8 8	
48	**Parámetros**	1 2 3	8 2 9	5 2 5	2,4 0,19 1,01	3	10	26	3	5 3 8	304,21
49	**Parámetros**	1 2 3	6 6 3	4 11 12	1,78 2,13 2,99	7	15	34	3	2 3 3	165,85
50	**Parámetros**	1 2 3	3 8 1	3 8 4	2,07 1,55 0,99	4	18	29	8	0 9 0	95,81
51	**Parámetros**	1 2 3	1 8 3	13 8 3	0,83 2,16 2,51	7	18	32	7	0 2 3	119,77
52	**Parámetros**	1 2 3	5 9 7	6 14 6	0,33 2,13 2,99	2	17	30	4	5 7 7	381,92
53	**Parámetros**	1 2 3	4 6 6	13 7 9	0,61 2,84 2	9	16	30	1	0 1 6	247,69
54	**Parámetros**	1 2 3	2 5 5	9 1 13	2,85 2,51 1,85	4	11	26	8	0 9 0	91,52
55	**Parámetros**	1 2 3	4 4 8	14 6 7	1,72 1,53 2,5	1	11	33	2	5 2 8	438,78
56	**Parámetros**	1 2 3	7 6 7	14 12 1	2,45 0,56 0,26	5	10	28	7	8 0 8	162,61
57	**Parámetros**	1 2 3	4 8 5	7 5 9	2,99 2,41 2,97	8	15	30	5	0 6 3	173,5
58	**Parámetros**	1 2 3	3 6 1	8 14 10	2,51 2,82 1,85	1	13	31	4	4 4 1	157,15
59	**Parámetros**	1 2 3	4 5 8	13 3 2	2,68 2,94 1,33	8	11	29	6	0 2 7	130,53
60	**Parámetros**	1 2 3	4 7 7	11 10 9	0,7 0,67 1,93	5	19	33	4	3 5 5	267,04
61	**Parámetros**	1 2 3	3 6 5	5 12 4	0,31 2,43 2,55	8	18	32	8	1 0 9	126,06
62	**Parámetros**	1 2 3	4 5 1	6 3 13	0,39 1,9 0,96	6	11	32	3	1 3 0	62

Modelo dinámico de cantidad económica de pedido con costo de preparación

		Periodo	Tasa de demanda	Costo de preparación de pedido	Costo unitario/ u. Tiempo de almacenamiento	Inventario inicial	Costo de pro primeras unidades	Costo de pro unidades siguientes	Unidades de primer costo de pro	CANTIDAD DE PEDIDO	COSTO MINIMO INVENTARIO
63	**Parámetros**	1 2 3	6 2 6	14 8 13	2,71 2,18 2,8	6	19	30	9	0 10 0	125,66
64	**Parámetros**	1 2 3	6 2 2	8 3 11	1,07 0,97 2,78	2	13	26	5	6 2 0	111,64
65	**Parámetros**	1 2 3	5 2 8	11 2 2	1,19 0,34 0,45	5	18	26	1	0 9 1	283,1
66	**Parámetros**	1 2 3	8 8 3	6 4 6	0,3 0,83 2,22	2	14	30	6	7 7 3	207,43
67	**Parámetros**	1 2 3	3 5 5	14 11 12	2,73 1,01 0,26	5	18	32	3	2 3 3	199,21
68	**Parámetros**	1 2 3	5 4 9	2 4 11	2,37 0,87 0,18	3	12	26	7	8 8 0	148,13
69	**Parámetros**	1 2 3	4 8 7	1 14 14	0,28 0,86 2,49	1	17	27	6	4 7 7	240,14
70	**Parámetros**	1 2 3	4 5 6	4 4 3	0,47 2,63 1,01	6	17	28	6	0 7 2	126,12
71	**Parámetros**	1 2 3	1 9 6	12 3 3	0,11 1,49 2,28	1	11	26	9	0 10 5	126,39
72	**Parámetros**	1 2 3	8 8 1	11 8 8	0,18 1,43 0,34	4	17	31	6	7 7 0	182,69
73	**Parámetros**	1 2 3	1 7 3	8 7 2	0,29 0,15 1,11	4	16	27	6	0 7 0	81,94
74	**Parámetros**	1 2 3	6 5 9	8 8 14	2,84 1,22 0,92	1	11	28	6	5 7 7	223,68
75	**Parámetros**	1 2 3	9 6 4	5 12 12	0,91 1,93 2,36	1	18	32	9	10 10 0	199,93
76	**Parámetros**	1 2 3	3 4 4	11 12 5	1,76 2,69 1,24	9	14	33	7	0 0 2	67,46
77	**Parámetros**	1 2 3	2 3 7	3 5 9	2,16 2,52 0,29	6	12	34	3	0 3 3	110,2
78	**Parámetros**					7	19	34	3		364,0,3

Modelo dinámico de cantidad económica de pedido con costo de preparación

		Periodo	Tasa de demanda	Costo de preparación de pedido	Costo unitario/ u. Tiempo de almacenamiento	Inventario inicial	Costo de pro primeras unidades	Costo de pro unidades siguientes	Unidades de primer costo de pro	CANTIDAD DE PEDIDO	COSTO MINIMO INVENTARIO
		1 2 3	9 6 6	4 5 3	2,67 2,17 2,31					4 4 6	
79	**Parámetros**	1 2 3	7 8 7	10 14 6	2,93 1,85 1,3	7	13	29	3	3 5 7	353,06
80	**Parámetros**	1 2 3	6 8 6	2 1 2	0,73 2,33 1,71	6	16	33	1	1 12 1	464,26
81	**Parámetros**	1 2 3	9 4 5	7 1 11	1,72 1,57 1,48	4	14	30	7	8 8 0	177,95
82	**Parámetros**	1 2 3	1 5 7	10 1 10	0,42 1,25 2,27	5	17	32	4	0 3 5	154,64
83	**Parámetros**	1 2 3	9 6 3	8 9 6	1,05 2,52 0,61	6	17	33	5	6 6 0	198,69
84	**Parámetros**	1 2 3	4 9 8	9 12 6	0,47 1,74 1,36	8	15	29	4	3 5 5	238,02
85	**Parámetros**	1 2 3	4 6 6	8 11 1	2,04 2,04 1,08	1	12	26	4	5 5 5	219,2
86	**Parámetros**	1 2 3	4 9 8	5 3 13	2,56 0,37 1,22	7	16	33	4	4 5 5	273,17
87	**Parámetros**	1 2 3	7 1 1	6 5 10	1,53 1,53 2,43	5	16	31	1	2 1 1	138,65
88	**Parámetros**	1 2 3	3 1 9	13 1 4	1,4 2,54 1,39	3	17	29	9	0 10 0	92,71
89	**Parámetros**	1 2 3	5 8 5	14 5 3	2,21 2,67 1,25	4	19	32	6	7 7 0	216,11
90	**Parámetros**	1 2 3	5 7 5	9 4 14	1,54 2,19 2,38	1	18	29	3	4 7 5	372,54
91	**Parámetros**	1 2 3	3 3 6	10 13 8	1,49 1,52 1,14	9	12	33	7	0 0 3	69,95
92	**Parámetros**	1 2 3	5 1 3	1 5 4	2,79 2,31 0,87	5	12	34	7	1 0 3	69,26

Modelo dinámico de cantidad económica de pedido con costo de preparación

		Periodo	Tasa de demanda	Costo de preparación de pedido	Costo unitario/ u. Tiempo de almacenamiento	Inventario inicial	Costo de pro primeras unidades	Costo de pro unidades siguientes	Unidades de primer costo de pro	CANTIDAD DE PEDIDO	COSTO MINIMO INVENTARIO
93	**Parámetros**	1 2 3	5 6 7	11 4 3	0,94 2,95 0,25	4	14	34	7	1 8 5	188,26
94	**Parámetros**	1 2 3	7 2 3	8 11 13	2,73 1,59 0,53	7	19	34	9	0 10 0	121,35
95	**Parámetros**	1 2 3	4 7 3	2 6 1	1,15 1,79 2,68	4	13	29	3	3 4 3	167,97
96	**Parámetros**	1 2 3	5 7 8	11 2 10	1,55 0,89 2,53	9	15	30	3	3 3 5	245,77
97	**Parámetros**	1 2 3	8 3 9	11 6 6	1,49 2,41 2,79	5	19	32	2	4 2 9	460,86
98	**Parámetros**	1 2 3	8 8 4	3 11 14	2,93 1,16 2,99	3	19	26	5	6 7 6	276,95
99	**Parámetros**	1 2 3	2 9 3	2 12 3	1,88 2,7 1,34	3	19	31	2	2 7 2	313,08
100	**Parámetros**	1 2 3	4 8 9	8 5 8	2,47 2,96 0,19	8	13	26	2	2 2 9	331,52

MODELOS DE INVENTARIOS PROBABILISTICOS

Modelo “probabilizado” de cantidad económica de pedido

Modelo "Probabilizado" de cantidad

Parámetros	Tiempo de entrega entre la colocación y la recepción de un pedido	Demanda promedio durante el tiempo de entrega	Desviación estándar de la demanda durante el tiempo de entrega	Probabilidad máxima admisible de que se agote la existencia durante el tiempo de entrega	CANTIDAD DE PEDIDO	PUNTO DE REORDEN
1	5	98	29	0,070	95,7	585,7
2	1	115	20	0,040	35,01	150,01
3	8	123	21	0,020	121,99	1105,99
4	9	171	14	0,010	97,71	1636,71
5	8	150	22	0,080	87,43	1295,43
6	7	125	30	0,030	149,28	1024,28
7	9	180	15	0,090	60,33	1680,33
8	4	186	13	0,090	34,86	778,86
9	7	91	11	0,010	67,7	704,7
10	4	194	29	0,070	85,6	861,6
11	7	165	26	0,010	160,03	1315,03
12	4	193	26	0,040	91,04	863,04
13	1	184	18	0,090	24,13	208,13
14	1	109	17	0,090	22,79	131,79
15	8	92	15	0,090	56,88	792,88
16	9	97	26	0,040	136,55	1009,55
17	4	135	16	0,040	56,02	596,02
18	1	136	17	0,040	29,76	165,76
19	8	163	11	0,010	72,38	1376,38
20	6	180	19	0,090	62,4	1142,4
21	7	190	18	0,030	89,57	1419,57
22	6	199	15	0,070	54,22	1248,22
23	7	129	18	0,050	78,33	981,33
24	7	182	21	0,010	129,25	1403,25
25	1	115	25	0,070	36,89	151,89
26	3	117	29	0,060	78,1	429,1
27	3	164	16	0,050	45,58	537,58
28	5	152	15	0,040	58,72	818,72
29	5	138	29	0,020	133,18	823,18
30	8	199	30	0,080	119,22	1711,22
31	6	150	13	0,010	74,08	974,08
32	5	139	13	0,040	50,89	745,89
33	4	129	13	0,060	40,42	556,42

Modelo "Probabilizado" de cantidad

Parámetros	Tiempo de entrega entre la colocación y la recepción de un pedido	Demanda promedio durante el tiempo de entrega	Desviación estándar de la demanda durante el tiempo de entrega	Probabilidad máxima admisible de que se agote la existencia durante el tiempo de entrega	CANTIDAD DE PEDIDO	PUNTO DE REORDEN
34	8	90	10	0,020	58,09	778,09
35	2	130	12	0,080	23,84	283,84
36	9	171	10	0,080	42,15	1581,15
37	1	120	26	0,020	53,4	173,4
38	1	103	22	0,050	36,19	139,19
39	9	123	29	0,060	135,27	1242,27
40	5	164	11	0,090	32,98	852,98
41	3	96	22	0,070	56,24	344,24
42	2	123	22	0,050	51,18	297,18
43	5	90	26	0,070	85,8	535,8
44	2	187	24	0,080	47,69	421,69
45	2	199	21	0,080	41,73	439,73
46	3	148	22	0,010	88,65	532,65
47	9	173	16	0,030	90,28	1647,28
48	8	118	30	0,020	174,24	1118,27
49	1	85	14	0,070	20,66	105,66
50	3	104	25	0,040	75,81	387,81
51	6	132	14	0,080	48,18	840,18
52	2	122	13	0,080	25,83	269,83
53	4	177	11	0,080	30,91	738,91
54	3	80	18	0,030	58,64	298,64
55	8	84	18	0,010	118,44	790,44
56	5	174	30	0,020	137,77	1007,77
57	3	122	23	0,010	92,68	458,68
58	6	194	23	0,060	87,59	1251,59
59	3	134	15	0,090	34,83	436,83
60	3	119	14	0,060	37,7	394,7
61	3	116	21	0,010	84,62	432,62
62	5	190	13	0,050	47,81	997,81
63	2	136	17	0,040	42,09	314,09
64	9	125	17	0,090	68,38	1193,38
65	9	151	27	0,090	108,6	1467,6
66	9	157	23	0,020	141,71	1554,71

Modelo "Probabilizado" de cantidad

Parámetros	Tiempo de entrega entre la colocación y la recepción de un pedido	Demanda promedio durante el tiempo de entrega	Desviación estándar de la demanda durante el tiempo de entrega	Probabilidad máxima admisible de que se agote la existencia durante el tiempo de entrega	CANTIDAD DE PEDIDO	PUNTO DE REORDEN
67	5	101	12	0,080	37,7	542,7
68	1	199	21	0,050	34,54	233,54
69	4	91	25	0,070	73,79	437,79
70	1	96	14	0,010	32,57	128,57
71	3	112	28	0,040	84,9	450,9
72	6	135	12	0,070	43,38	853,38
73	3	182	20	0,010	80,59	626,59
74	9	117	29	0,030	163,63	1216,63
75	4	122	18	0,080	50,58	538,58
76	1	110	17	0,060	26,43	136,43
77	5	186	17	0,050	62,53	992,53
78	8	80	28	0,040	138,65	778,65
79	9	152	14	0,050	69,08	1437,08
80	3	185	15	0,010	60,44	615,44
81	5	172	23	0,040	90,04	950,04
82	4	100	12	0,070	35,42	435,42
83	2	117	10	0,030	26,6	260,6
84	5	95	26	0,040	101,78	576,78
85	3	142	18	0,060	48,47	474,47
86	6	113	14	0,050	56,41	734,41
87	7	199	17	0,070	66,38	1459,38
88	5	120	13	0,010	67,62	667,62
89	7	114	19	0,050	82,69	880,69
90	3	103	29	0,080	70,58	379,58
91	8	134	16	0,030	85,12	1157,12
92	8	199	30	0,070	125,23	1717,23
93	4	118	27	0,020	110,9	582,9
94	3	149	12	0,070	30,67	477,67
95	8	94	25	0,020	145,22	897,22
96	2	140	21	0,030	55,86	335,86
97	1	93	13	0,010	30,24	123,24
98	2	126	25	0,040	61,9	313,9
99	6	135	19	0,010	108,27	918,27

SIMULACIÓN DE DATOS

Modelo "Probabilizado" de cantidad

Parámetros	Tiempo de entrega entre la colocación y la recepción de un pedido	Demanda promedio durante el tiempo de entrega	Desviación estándar de la demanda durante el tiempo de entrega	Probabilidad máxima admisible de que se agote la existencia durante el tiempo de entrega	CANTIDAD DE PEDIDO	PUNTO DE REORDEN
100	1	176	29	0,020	59,56	235,56

Modelo probabilístico de cantidad económica de pedido

SIMULACIÓN DE DATOS

Modelo probabilístico de cantidad económica de pedido

Parámetros	Demanda esperada por unidad de tiempo	Costo de preparación por pedido	Costo de almacenamiento por unidad de inventario y por unidad de tiempo	Costo de faltante por unidad de inventario	Función de distribución de probabilidades de la demanda x durante el tiempo de entrega	CANTIDA DE PEDIDO	PUNTO DE REORDEN
1	1000	150	3	15	50	317,8207	46,82
2	2000	200	2	20	25	448,88	24,43
3	1000	100	2	10	100	319,438	93,611
4	1125	110	4	12	25	249,67	23,1505
5	825	67	3	10	35	193,1967	32,5411
6	838	133	2	12	60	335,856	55,99
7	870	66	7	10	30	129,65	26,869
8	1008	55	13	20	75	94,667	70,42
9	1109	124	9	19	20	175,56	18,5
10	885	112	8	16	28	158,67	57,54
11	1067	137	3	9	75	315,89	38,46
12	1180	131	2	8	35	394,632	32,073
13	1026	117	4	12	15	245,59	13,803
14	965	131	4	14	36	252,761	33,3
15	966	131	2	7	45	358,124	67,174

Modelo probabilístico de cantidad económica de pedido

Parámetros	Demanda esperada por unidad de tiempo	Costo de preparación por pedido	Costo de almacenamiento por unidad de inventario y por unidad de tiempo	Costo de faltante por unidad de inventario	Función de distribución de probabilidades de la demanda x durante el tiempo de entrega	CANTIDA DE PEDIDO	PUNTO DE REORDEN
16	1163	86	9	24	65	150,63	61,842
17	1023	97	6	11	35	183,59	48,34
18	1023	97	6	11	35	182,22	18,824
19	925	142	2	9	40	364,17	94,95
20	801	79	9	15	45	124,55	15,032
21	1049	135	7	13	48	203,673	42,981
22	873	101	2	5	68	301,67	58,599
23	1055	110	4	10	55	243,43	49,923
24	855	83	3	10	80	220,62	73,8
25	870	124	2	9	15	329,081	13,739
26	810	108	7	17	13	158,62	11,951
27	876	106	19	29	10	98,865	9,992
28	1158	136	19	39	200	134,54	11,207
29	984	123	9	21	90	167,312	23,65
30	854	132	23	43	26	99,823	57,56
31	813	125	30	90	120	84,413	16,61
32	802	118	7	27	88	166,825	33,254
33	948	54	29	60	76	60,604	25,002
34	937	136	4	19	10	270,551	31,504
35	845	55	3	7	220	220,706	7,856
36	1086	66	43	93	15	57,924	14,63
37	914	118	24	84	6	94,892	5,822
38	1048	51	93	104	85	35,108	13,846
39	135	90	9	60	21	52,564	19,772
40	894	129	31	99	76	87,429	36,409
41	1136	93	24	36	63	95,612	25,522
42	1125	100	63	100	360	66,828	4,774
43	1026	117	5	11	200	219,157	198,917
44	1051	82	26	36	17	478,586	57,447
45	1067	130	90	110	131	58,537	9,503
46	890	111	26	80	113	89,03	23,442

SIMULACIÓN DE DATOS

Modelo probabilístico de cantidad económica de pedido

Parámetros	Demanda esperada por unidad de tiempo	Costo de preparación por pedido	Costo de almacenamiento por unidad de inventario y por unidad de tiempo	Costo de faltante por unidad de inventario	Función de distribución de probabilidades de la demanda x durante el tiempo de entrega	CANTIDA DE PEDIDO	PUNTO DE REORDEN
47	918	139	34	86	213	219,542	104,096
48	1025	132	2	4	316	399,914	5,214
49	1055	110	4	10	55	185,668	5,247
50	855	83	3	10	80	217,71	26,087

Modelo sin preparación

SIMULACIÓN DE DATOS

Modelo sin preparación

Parámetros	Costo de compra por unidad	Costo de almacenamiento por unidad conservada durante el periodo	Penalización por unidad faltante durante el periodo	Cantidad a la mano, antes de hacer un pedido	Media de la demanda	Desviación estándar de la demanda	CANTIDAD DE PEDIDO	PUNTO DE REORDEN
1	30	25	45	0	300	20	284	3063,24
2	46	16	80	0	255	30	244	2644,86
3	41	33	63	0	279	43	247	2925,97
4	95	15	70	2	246	41	200	2589,65
5	31	12	85	0	330	17	332	3353,75
6	70	10	69	2	276	41	200	2889,65
7	42	29	96	1	284	28	279	2928,54
8	26	39	71	1	309	37	300	3207,004
9	52	18	50	3	261	45	200	2752,3
10	24	37	51	3	267	31	251	2768,03
11	45	46	82	0	218	20	207	2243,24
12	20	16	86	0	252	34	265	2627,51
13	9	37	57	2	273	15	273	2777,43
14	25	30	94	1	261	31	265	2708,03

SIMULACIÓN DE DATOS

Modelo sin preparación

Parámetros	Costo de compra por unidad	Costo de almacenamiento por unidad conservada durante el periodo	Penalización por unidad faltante durante el periodo	Cantidad a la mano, antes de hacer un pedido	Media de la demanda	Desviación estándar de la demanda	CANTIDAD DE PEDIDO	PUNTO DE REORDEN
15	94	31	91	1	328	42	200	3412,81
16	30	41	98	2	253	33	252	2634,35
17	93	20	91	3	301	26	200	3092,21
18	20	37	77	3	300	24	300	3075,89
19	39	24	82	0	336	28	329	3448,54
20	26	41	70	1	232	19	227	2380,08
21	17	12	58	2	338	16	341	3430,59
22	46	19	82	3	277	19	270	2830,08
23	65	32	83	3	223	39	200	2353,32
24	30	19	86	1	275	23	277	2822,73
25	36	37	90	2	339	45	331	3532,3
26	72	19	88	2	236	37	200	2477,004
27	33	44	71	3	337	36	321	3483,84
28	51	10	89	3	283	18	278	2886,92
29	73	19	56	0	280	40	200	2926,49
30	18	17	99	2	324	43	346	3375,97
31	93	35	61	0	320	42	200	3332,81
32	38	22	87	2	266	15	264	2707,43
33	59	39	48	3	268	33	200	2784,35
34	7	32	53	0	336	43	340	3495,97
35	67	26	88	3	247	21	228	2536,4
36	37	34	74	2	336	37	321	3477,004
37	74	10	90	0	338	33	305	3484,35
38	37	40	88	1	230	17	226	2353,75
39	66	50	76	1	330	19	303	3360,08
40	66	26	80	2	343	33	306	3534,35
41	16	33	58	1	307	32	304	3171
42	31	35	68	2	325	34	313	3357,51
43	21	48	56	2	284	23	274	2912,73
44	68	44	96	2	247	23	228	2548,73

SIMULACIÓN DE DATOS

Modelo sin preparación

Parámetros	Costo de compra por unidad	Costo de almacenamiento por unidad conservada durante el periodo	Penalización por unidad faltante durante el periodo	Cantidad a la mano, antes de hacer un pedido	Media de la demanda	Desviación estándar de la demanda	CANTIDAD DE PEDIDO	PUNTO DE REORDEN
45	80	28	68	3	290	30	200	2994,86
46	84	38	66	3	336	18	200	3416,92
47	11	17	82	0	235	18	245	2406,92
48	71	26	96	1	316	23	297	3232,73
49	26	12	58	3	283	15	281	2877,43
50	89	43	55	1	235	34	200	2457,51
51	58	11	86	2	342	28	326	3508,54
52	8	41	81	2	313	44	324	3269,14
53	8	35	61	0	317	36	322	3283,84
54	62	10	60	0	324	24	200	3315,89
55	88	35	83	1	338	28	200	3468,5437
56	4	41	70	0	223	25	229	2309,05
57	83	19	67	0	346	45	200	3602,3
58	90	15	81	2	262	44	200	2759,14
59	76	34	98	3	338	16	323	3430,59
60	42	22	48	3	247	25	213	2549,05
61	29	46	91	3	234	22	231	2409,57
62	69	44	98	1	281	20	264	2873,24
63	19	36	84	0	330	32	333	3401,19
64	40	36	50	0	232	24	303	3395,89
65	59	48	64	1	236	21	200	2426,4
66	77	38	82	1	284	35	223	2950,67
67	15	17	94	3	242	21	254	2486,4
68	91	22	52	1	240	19	200	2460,08
69	35	46	99	2	242	37	237	2537,004
70	85	30	58	2	305	33	200	3154,35
71	34	22	65	0	254	42	239	2672,81
72	89	13	98	0	294	35	245	3050,67
73	69	43	90	0	313	34	279	3237,51
74	96	13	73	1	286	26	200	2942

SIMULACIÓN DE DATOS

Modelo sin preparación

Parámetros	Costo de compra por unidad	Costo de almacenamiento por unidad conservada durante el periodo	Penalización por unidad faltante durante el periodo	Cantidad a la mano, antes de hacer un pedido	Media de la demanda	Desviación estándar de la demanda	CANTIDAD DE PEDIDO	PUNTO DE REORDEN
75	53	49	99	2	312	39	293	3243,32
76	90	44	75	0	299	22	200	3059,57
77	34	21	63	2	286	36	272	2973,84
78	95	23	49	1	261	39	200	2733,32
79	90	38	50	2	269	32	200	2791,19
80	78	11	62	3	292	39	200	3043,32
81	57	40	78	1	303	21	284	3096,4
82	81	48	86	1	319	30	266	3284,86
83	98	33	85	0	247	19	200	2530,08
84	100	46	49	1	280	20	200	2863,24
85	65	20	70	2	206	21	200	2126,4
86	76	50	94	3	330	31	294	3398,03
87	16	16	63	3	241	42	251	2542,81
88	30	40	89	2	338	23	336	3452,73
89	64	31	81	1	274	16	258	2790,59
90	5	26	50	1	257	28	264	2658,54
91	81	26	52	0	200	41	200	2129,65
92	90	48	55	3	208	33	200	2184,35
93	53	38	72	1	221	27	200	2295,38
94	35	38	73	0	227	25	217	2349,05
95	80	20	89	3	259	16	237	2640,59
96	31	40	81	3	200	45	200	2142,3
97	29	45	92	1	231	17	229	2363,75
98	71	45	46	5	246	32	200	2561,19
99	36	44	59	1	313	38	284	3250,16
100	86	25	47	3	306	40	200	3186,49

Modelo con preparación (política s-S)

SIMULACIÓN DE DATOS

Modelo con preparación (política s-S)

Parámetros	Cantidad a la mano, antes de hacer un pedido	Costo de preparación de pedido	Costo de compra por unidad	Costo de almacenamiento por unidad conservada durante el periodo	Penalización por unidad faltante durante el periodo	Demanda máxima (D_max)	S	s1	s2	Cantidad a pedir
1	**5**	24,72	2,04	2,86	18,3	94	7,68	17,27	-1,91	No pedir
2	**29**	63,19	4,54	0,78	12,2	26	15,34	31,63	0,28	No pedir
3	**21**	46,5	8,61	3,81	14,29	20	6,27	16,47	-3,85	No pedir
4	**6**	30,32	3,95	3,08	6,46	23	6,05	18,58	-6,03	No pedir
5	**22**	59,36	5,29	4,88	12,96	12	5,15	14,13	-3,77	No pedir
6	**30**	78,71	4,27	1,76	15,9	73	48,07	74,07	22,84	Pedir
7	**24**	62,23	4,42	2,67	6,1	70	13,4	45,22	-18,13	No pedir
8	**3**	26,39	4,3	7,3	13,13	10	4,32	9,41	-0,76	No pedir
9	**13**	58,77	1,37	3,09	6,9	50	27,67	56,45	4,98	Pedir
10	**15**	26,97	8,26	3,17	13,13	63	18,82	33,33	4,41	Pedir
11	**6**	48,4	8,69	4,95	16,3	60	21,48	38,03	4,96	Pedir
12	**3**	17,19	4,69	4,36	26,27	75	52,84	62,01	43,66	Pedir
13	**13**	40,41	3,7	3,96	9,86	10	4,45	17,94	-7,01	No pedir
14	**13**	25,2	5,74	3,67	14,14	13	6,13	12,28	0,06	No pedir
15	**16**	23,7	8,13	2,39	10	28	4,22	14,62	-6,12	No pedir
16	**10**	57,69	4,53	3,06	11,14	68	31,65	57,1	8,99	Pedir
17	**19**	76,04	8,06	1,56	9,53	100	13,25	50,54	-23,81	No pedir
18	**26**	48,26	1,94	0,83	1,82	59	50,41	67,81	33,18	Pedir
19	**22**	84,07	3,14	2,08	13,26	55	36,28	64,48	13,35	Pedir
20	**21**	50,66	5,45	3,88	28	54	38,19	52,11	25,64	Pedir
21	**50**	84,07	6,73	0,64	25	97	69,11	95,72	44,81	Pedir
22	**16**	92,5	9,78	0,57	31,72	10	6,79	14,39	-0,76	No pedir
23	**8**	17,47	3,6	4,62	4,58	16	1,7	9,53	-6,11	No pedir
24	**29**	91,56	0,74	3,61	15,13	3	7,71	9,92	-5,29	No pedir
25	**11**	36,45	8,81	4,93	19,76	94	4,43	82,52	0,87	No pedir
26	**19**	21	0,1	2	6,5	5	3,76	75,43	-0,153	No pedir
27	**7**	52	9,8	1,7	16	35	15,17	26,89	-2,37	No pedir

SIMULACIÓN DE DATOS

Modelo con preparación (política s-S)

Parámetros	Cantidad a la mano, antes de hacer un pedido	Costo de preparación de pedido	Costo de compra por unidad	Costo de almacenamiento por unidad conservada durante el periodo	Penalización por unidad faltante durante el periodo	Demanda máxima (D_max)	S	s1	s2	Cantidad a pedir
28	**21**	37	8	1,8	11	52	12,18	29,52	-5,15	No pedir
29	**14**	84	5	4,4	7	37	6,49	30,33	-16,99	No pedir
30	**15**	18	8	0,5	22	89	60,76	73,49	48,45	Pedir
31	**10**	98	6	0,1	12,5	65	33,53	65,33	1,73	No pedir
32	**2**	63	5	3,5	19	39	10,1	44,81	3,79	Pedir
33	**8**	71	9	1,5	20	96	50,28	74,32	23,91	Pedir
34	**18**	10	1,5	1,5	8	60	45,16	56,95	33,37	Pedir
35	**13**	19	7	1,8	19,5	5	2,93	5,92	-0,05	No pedir
36	**21**	44	7	1,3	30	85	62,46	77,92	47,003	Pedir
37	**8**	28	6	4,1	33	43	31,29	39,35	23,23	Pedir
38	**17**	86	8	2	9,4	45	5,53	31,59	-20,53	No pedir
39	**9**	15	4,5	1,2	13	62	37,1	48,56	25,66	Pedir
40	**15**	48	2	1,9	7	85	47,75	78,036	17,47	Pedir
41	**10**	49	1,3	2,3	17	29	23,59	50,62	-344	No pedir
42	**11**	60	6	3,5	15	84	40,86	64,21	17,52	Pedir
43	**24**	81	8	1,4	18	17	8,76	20,67	-3,15	No pedir
44	**5**	93	7	0,4	11,5	75	28,36	-3,85	9,53	Pedir
45	**19**	88	1,8	3,2	7	10	5,09	18,23	-8,03	No pedir
46	**6**	31	7	1,1	72	14	5,34	13,48	-2,79	No pedir
47	**26**	31	20	21	26	18	3,84	10,145	-2,45	No pedir
48	**17**	11	8,4	1,3	10,4	48	8,205	17,7	-1,29	No pedir
49	**5**	51	3	2,5	12	55	34,13	53,81	14,46	Pedir
50	**21**	58	4	0,4	12	92	59,35	88,69	30,01	Pedir

Modelos de varios periodos

SIMULACIÓN DE DATOS

Modelos de varios periodos

Parámetros	Nivel inicial de inventario	Costo de penalización por unidad y por mes	Factor de descuento	Precio de venta por unidad	Precio de compra por unidad	Costo de almacenamiento por unidad y por mes	Demanda máxima (D_max)	NIVEL ÓPTIMO DE INVENTARIO	CANTIDAD A PEDIR
1	10	3	0,8	2	1	0,1	10	13,71	Pedir
2	1	67	0,6	6	44	0,3	39	3,47	Pedir
3	8	85	0,7	16	29	0,5	18	12,54	Pedir
4	7	87	0,8	20	24	0,5	45	39,25	Pedir
5	4	54	0,3	22	49	0,2	34	9,23	Pedir
6	11	41	0,3	45	46	0,4	50	27,23	Pedir
7	8	55	0,6	6	30	0,6	10	2,86	No pedir
8	11	31	0,3	21	38	0,1	25	4,86	No pedir
9	13	35	0,7	71	58	0,5	20	20,11	Pedir
10	10	63	0,4	5	41	0,4	11	2,09	No pedir
11	3	61	0,4	2	33	0,7	40	11,19	Pedir
12	7	92	0,4	8	36	0,2	43	23,41	Pedir
13	12	60	0,5	11	37	0,1	19	6,08	No pedir
14	9	53	0,8	13	25	0,3	35	19,66	Pedir
15	4	73	0,5	12	43	0,5	28	9,33	Pedir
16	8	87	0,8	17	52	0,3	41	10,85	Pedir
17	12	54	0,4	24	41	0,7	38	16,61	Pedir
18	10	46	0,7	24	40	0,4	38	13,33	Pedir
19	5	41	0,8	18	39	0,5	30	2,13	No pedir
20	10	31	0,6	88	43	0,6	30	46,26	Pedir
21	7	40	0,6	19	28	0,8	44	23,27	Pedir
22	11	79	0,8	9	42	0,7	49	11,78	No pedir
23	5	52	0,6	9	29	0,5	48	17,11	Pedir
24	9	56	0,7	55	48	0,1	26	24,32	Pedir
25	6	36	0,5	6	22	0,6	18	5,45	No pedir
26	8	44	0,7	10	27	0,2	15	4,80	No pedir

Modelos de varios periodos

Parámetros	Nivel inicial de inventario	Costo de penalización por unidad y por mes	Factor de descuento	Precio de venta por unidad	Precio de compra por unidad	Costo de almacenamiento por unidad y por mes	Demanda máxima (D_max)	NIVEL ÓPTIMO DE INVENTARIO	CANTIDAD A PEDIR
27	10	42	0,7	23	31	0,2	38	21,98	Pedir
28	11	99	0,3	9	48	0,7	44	20,05	Pedir
29	6	83	0,5	21	28	0,7	44	33,86	Pedir
30	7	86	0,3	8	24	0,4	21	14,88	Pedir
31	11	91	0,5	22	22	0,2	16	14,25	Pedir
32	11	75	0,5	56	58	0,3	31	21,61	Pedir
33	3	42	0,7	53	48	0,5	29	25,08	Pedir
34	10	37	0,7	36	54	0,5	11	1,46	No pedir
35	9	54	0,4	8	24	0,5	37	19,72	Pedir
36	3	65	0,7	4	35	0,6	35	6,44	Pedir
37	5	64	0,3	23	48	0,7	32	12,48	Pedir
38	4	87	0,3	20	22	0,8	49	40,62	Pedir
39	5	100	0,6	22	58	0,1	15	5,84	Pedir
40	6	39	0,3	16	35	0,7	21	5,90	No pedir
41	4	73	0,4	18	23	0,2	37	29,07	Pedir
42	10	48	0,4	13	36	0,1	14	3,96	No pedir
43	6	48	0,5	26	48	0,1	11	2,70	No pedir
44	8	35	0,5	13	25	0,1	49	20,02	Pedir
45	3	48	0,5	90	88	0,4	10	5,46	Pedir
46	6	98	0,3	18	41	0,4	47	28,84	Pedir
47	7	78	0,7	19	54	0,1	18	3,97	No pedir
48	8	37	0,8	3	50	0,2	35	-44,07	No pedir
49	9	48	0,5	91	58	0,8	48	49,63	Pedir
50	9	45	0,3	10	28	0,2	36	14,90	Pedir
51	7	75	0,7	22	43	0,3	41	19,67	Pedir
52	3	67	0,6	4	53	0,4	20	-3,30	No pedir
53	4	50	0,6	4	54	0,7	27	-15,49	No pedir
54	12	37	0,6	8	55	0,7	10	-9,34	No pedir
55	12	82	0,3	15	32	0,3	15	9,68	No pedir

Modelos de varios periodos

Parámetros	Nivel inicial de inventario	Costo de penalización por unidad y por mes	Factor de descuento	Precio de venta por unidad	Precio de compra por unidad	Costo de almacenamiento por unidad y por mes	Demanda máxima (D_max)	NIVEL ÓPTIMO DE INVENTARIO	CANTIDAD A PEDIR
56	9	88	0,4	6	27	0,1	10	6,39	No pedir
57	8	53	0,6	12	47	0,4	21	-1,08	No pedir
58	3	87	0,7	6	56	0,5	37	0,83	No pedir
59	13	58	0,8	22	44	0,8	49	14,27	Pedir
60	10	77	0,4	17	27	0,2	18	12,97	Pedir
61	5	100	0,3	12	51	0,1	50	22,72	Pedir
62	10	47	0,5	25	59	0,6	29	-1,93	No pedir
63	8	55	0,6	24	24	0,1	39	33,15	Pedir
64	8	92	0,7	8	42	0,5	30	10,81	Pedir
65	4	47	0,8	11	33	0,4	17	2,54	No pedir
66	9	83	0,5	18	36	0,4	28	16,97	Pedir
67	7	99	0,6	10	52	0,4	20	6,15	No pedir
68	3	43	0,8	18	20	0,5	17	14,22	Pedir
69	6	84	0,4	3	38	0,1	37	15,08	Pedir
70	6	51	0,5	8	33	0,2	31	7,58	Pedir
71	5	95	0,3	3	49	0,2	23	8,32	Pedir
72	11	77	0,4	13	26	0,5	37	25,51	Pedir
73	3	88	0,3	11	59	0,5	46	12,24	Pedir
74	3	31	0,6	18	44	0,4	43	-11,81	No pedir
75	12	89	0,8	11	45	0,8	23	6,95	No pedir
76	5	46	0,4	12	60	0,5	35	-13,82	No pedir
77	8	61	0,8	18	56	0,7	47	-5,33	No pedir
78	3	89	0,7	11	28	0,1	16	10,41	Pedir
79	12	98	0,4	15	47	0,6	46	22,74	Pedir
80	9	62	0,8	25	28	0,8	12	10,02	Pedir
81	11	56	0,5	12	50	0,5	47	-0,75	No pedir
82	13	80	0,7	11	25	0,2	34	22,88	Pedir
83	6	94	0,5	21	52	0,7	10	4,52	No pedir
84	9	55	0,6	2	35	0,1	44	1,73	No pedir

SIMULACIÓN DE DATOS

Modelos de varios periodos

Parámetros	Nivel inicial de inventario	Costo de penalización por unidad y por mes	Factor de descuento	Precio de venta por unidad	Precio de compra por unidad	Costo de almacenamiento por unidad y por mes	Demanda máxima (D_max)	NIVEL ÓPTIMO DE INVENTARIO	CANTIDAD A PEDIR
85	7	59	0,3	18	52	0,5	38	7,80	No pedir
86	9	57	0,4	13	53	0,5	36	0,55	No pedir
87	12	56	0,6	14	38	0,5	18	5,10	No pedir
88	3	40	0,4	18	52	0,8	15	-2,21	No pedir
89	8	36	0,8	18	33	0,7	12	2,68	No pedir
90	3	49	0,8	8	31	0,2	26	3,89	No pedir
91	12	55	0,7	5	32	0,1	37	5,95	No pedir
92	6	71	0,6	13	56	0,4	29	0,83	No pedir
93	3	50	0,6	12	42	0,7	12	0,43	No pedir
94	13	74	0,8	25	52	0,1	28	8,99	No pedir
95	6	91	0,5	25	56	0,7	42	17,94	Pedir
96	10	92	0,8	23	51	0,2	18	7,74	No pedir
97	6	66	0,6	4	55	0,7	27	-6,17	No pedir
98	4	81	0,4	5	44	0,2	41	12,86	Pedir
99	9	69	0,8	24	35	0,7	27	17,83	Pedir
100	11	58	0,4	9	54	0,5	33	-2,58	No pedir

Anexo 9. Códigos árbol de decisiones y redes neuronales

https://github.com/LuisStiven1313/Codigo

Anexo 10. Código muestreo *bootstrap*

```
import numpy as np
```

```
# Datos empíricos originales
empirical_data = np.array([144, 2900, 3205, 250, 96, 173, 384, 600, 61, 30])

# Número de muestras bootstrap necesarias para alcanzar un total de 100 datos
n_bootstrap_samples = 100 - len(empirical_data)

# Generar muestras bootstrap seleccionando datos al azar con reemplazo
bootstrap_samples = np.random.choice(empirical_data, n_bootstrap_samples, replace=True)

# Combinar los datos empíricos originales con las muestras bootstrap para obtener 100 datos
combined_data = np.concatenate((empirical_data, bootstrap_samples))

# Mostrar el resultado
print(combined_data)
```

Anexo 11. Código normalidad y *T-student*

T-STUDENT

```
[2] !pip install pingouin
```

Mostrar salida oculta

```
[3] import pandas as pd
    import numpy as np
    import matplotlib.pyplot as plt
    import seaborn as sns
    from scipy import stats
    import pingouin as pg
```

```
# Descarga de datos
# ==============================================================================
url = ('https://raw.githubusercontent.com/JoaquinAmatRodrigo/' +
       'Estadistica-machine-learning-python/master/data/births.csv')
datos = pd.read_csv(url, sep=',')
datos.head(4)


datos = pd.read_excel("/content/drive/MyDrive/T-STUDENT.xlsx")
datos

empirico = datos["empirico"]
programa = datos["programa"]

print(empirico)
print(programa)
```

Mostrar salida oculta

```
# Gráficos de distribución
# ============================================================================
fig, axs = plt.subplots(2, 2, figsize=(10, 7))

#peso_smokers = datos.loc[datos.smoke == 'smoker', 'weight']
# Valores de la media (mu) y desviación típica (sigma) de cada grupo
mu, sigma = stats.norm.fit(empirico)

# Valores teóricos de la normal en el rango observado
x_hat = np.linspace(min(empirico), max(empirico), num=100)
y_hat = stats.norm.pdf(x_hat, mu, sigma)

# Gráfico distribución
axs[0, 0].plot(x_hat, y_hat, linewidth=2, label='normal')
axs[0, 0].hist(x=empirico, density=True, bins=20, color="#3182bd", alpha=0.5)
axs[0, 0].plot(empirico, np.full_like(empirico, -0.01), '|k', markeredgewidth=1)
axs[0, 0].set_title('Distribución datos empirico')
axs[0, 0].set_xlabel('empirico')
axs[0, 0].set_ylabel('Densidad de probabilidad')
axs[0, 0].legend()

# Gráfico distribución qq-plot
pg.qqplot(empirico, dist='norm', ax=axs[0, 1])

#peso_nonsmokers = datos.loc[datos.smoke == 'nonsmoker', 'weight']
mu, sigma = stats.norm.fit(programa)
x_hat = np.linspace(min(programa), max(programa), num=100)
y_hat = stats.norm.pdf(x_hat, mu, sigma)
axs[1, 0].plot(x_hat, y_hat, linewidth=2, label='normal')
axs[1, 0].hist(x=programa, density=True, bins=20, color="#3182bd", alpha=0.5)
axs[1, 0].plot(programa, np.full_like(programa, -0.01), '|k', markeredgewidth=1)
axs[1, 0].set_title('Distribución datos programa')
axs[1, 0].set_xlabel('programa')
axs[1, 0].set_ylabel('Densidad de probabilidad')
axs[1, 0].legend()

pg.qqplot(programa, dist='norm', ax=axs[1, 1])
plt.tight_layout();
```

```
[10] # Test de normalidad Shapiro-Wilk
     # ============================================================
     print(pg.normality(data=empirico))
     pg.normality(data=programa)
```

```
                W         pval  normal
empirico  0.610518  6.848007e-15   False
```

	W	pval	normal
programa	0.971913	0.031024	False

T-TEST

```
[11] # Test para datos independientes (p-value, intervalos de confianza)
     # ============================================================================

     pg.ttest(x=empirico, y=programa, correction=False)
```

	T	dof	alternative	p-val	CI95%	cohen-d	BF10	power
T-test	5.031272	198	two-sided	0.000001	[368.24, 842.99]	0.711529	1.277e+04	0.998843