UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE ECONOMÍA Y EMPRESA
CARRERA DE NEGOCIOS INTERNACIONALES

TEMA:

Aplicación del machine learning para la clasificación y predicción e importación de repuestos tipo crítico para automotores.

AUTOR (AS):
Chica Vega, María Renata
Sucuzhanay Ceron, Michela Francesca

Trabajo de titulación previo a la obtención del título de
LICENCIADO EN NEGOCIOS INTERNACIONALES

TUTOR:
Ing. Carrera Buri, Félix Miguel, Mgs.

Guayaquil, Ecuador
13 de febrero del 2026

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE ECONOMÍA Y EMPRESA
CARRERA DE NEGOCIOS INTERNACIONALES

CERTIFICACIÓN

Certificamos que el presente trabajo de titulación fue realizado en su totalidad por **Chica Vega, María Renata y Sucuzhanay Ceron, Michela Francesca** como requerimiento para la obtención del título de **Licenciados en Negocios Internacionales.**

TUTOR

f._____
Ing. Carrera Buri, Félix Miguel, Mgs.

DIRECTOR DE LA CARRERA

f._____

Ing. Hurtado Cevallos, Gabriela Elizabeth, Mgs.

Guayaquil, a los 13 días del mes de febrero del año 2026

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE ECONOMÍA Y EMPRESA
CARRERA DE NEGOCIOS INTERNACIONALES

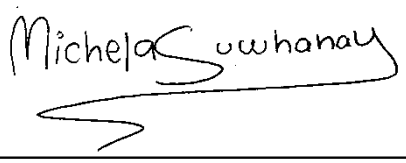
DECLARACIÓN DE RESPONSABILIDAD

Yo, **Chica Vega, María Renata**
Sucuzhanay Ceron, Michela Francesca

DECLARO QUE:

El Trabajo de Titulación, **Aplicación del machine learning para la clasificación y predicción e importación de repuestos tipo crítico para automotores,** previo a la obtención del título de Licenciados en Negocios Internacionales, ha sido desarrollado respetando derechos intelectuales de terceros conforme las citas que constan en el documento, cuyas fuentes se incorporan en las referencias o bibliografías. Consecuentemente este trabajo es de mi total autoría. En virtud de esta declaración, me responsabilizo del contenido, veracidad y alcance del Trabajo de Titulación referido.

Guayaquil, a los 13 días del mes de febrero del año 2026
AUTORES:



Chica Vega, María Renata | **Sucuzhanay Ceron, Michela Francesca**

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE ECONOMÍA Y EMPRESA
CARRERA DE NEGOCIOS INTERNACIONALES

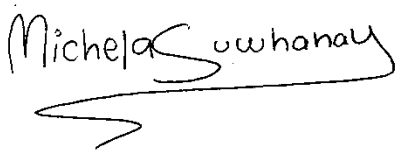
AUTORIZACIÓN

Yo, **Chica Vega, María Renata**
Sucuzhanay Ceron, Michela Francesca

Autorizo a la Universidad Católica de Santiago de Guayaquil a la publicación en la biblioteca de la institución del Trabajo de Titulación, **Aplicación del machine learning para la clasificación y predicción e importación de repuestos tipo crítico para automotores**, cuyo contenido, ideas y criterios son de mi exclusiva responsabilidad y total autoría.

Guayaquil, a los 13 días del mes de febrero del año 2026

AUTORES:



Chica Vega, María Renata

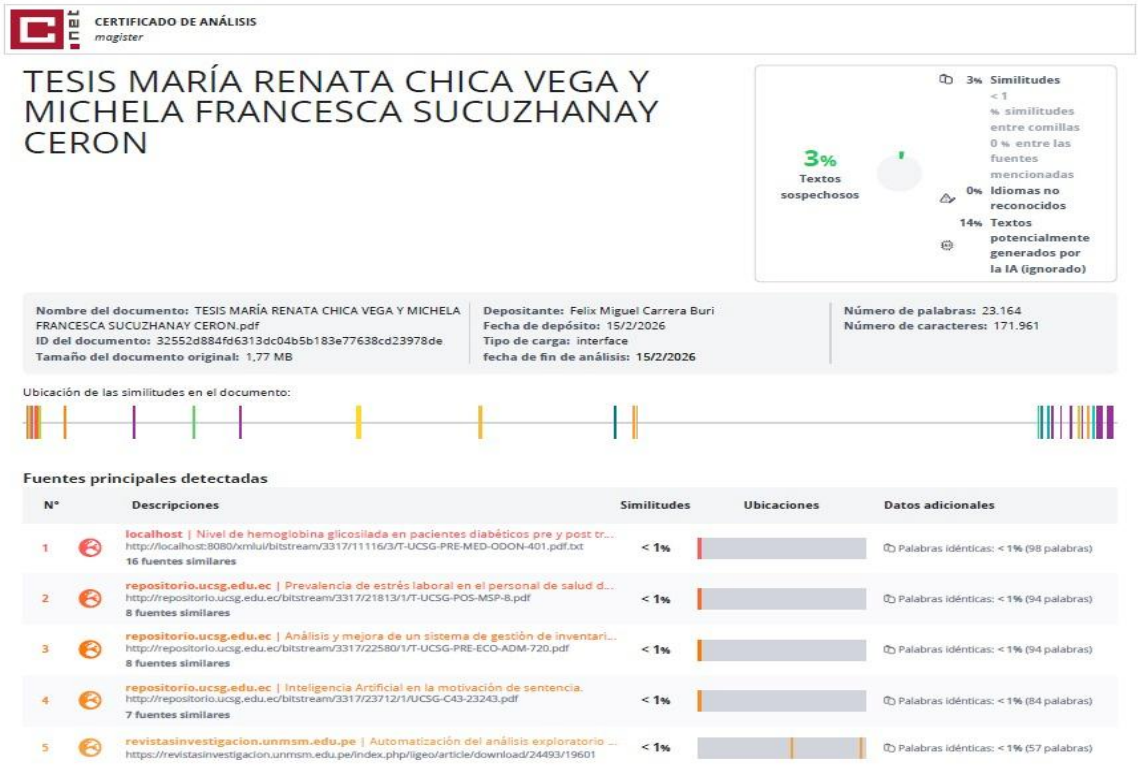
Sucuzhanay Ceron, Michela Francesca

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE ECONOMÍA Y EMPRESA

CARRERA DE NEGOCIOS INTERNACIONALES

REPORTE COMPILATIO



CERTIFICADO DE ANÁLISIS
magister

TESIS MARÍA RENATA CHICA VEGA Y MICHELA FRANCESCA SUCUZHANAY CERON

3% Textos sospechosos

3% Similitudes
< 1 % similitudes entre comillas
0 % entre las fuentes mencionadas
0% Idiomas no reconocidos
14% Textos potencialmente generados por la IA (ignorado)

Nombre del documento: TESIS MARÍA RENATA CHICA VEGA Y MICHELA FRANCESCA SUCUZHANAY CERON.pdf
ID del documento: 32552d884fd6313dc04b5b183e77638cd23978de
Tamaño del documento original: 1,77 MB

Depositante: Felix Miguel Carrera Buri
Fecha de depósito: 15/2/2026
Tipo de carga: interface
fecha de fin de análisis: 15/2/2026

Número de palabras: 23.164
Número de caracteres: 171.961

Ubicación de las similitudes en el documento:

Fuentes principales detectadas

Nº	Descripciones	Similitudes	Ubicaciones	Datos adicionales
1	localhost | Nivel de hemoglobina glicosilada en pacientes diabéticos pre y post tr... http://localhost:8080/xmlui/bitstream/3317/11116/3/T-UCSG-PRE-MED-ODON-401.pdf.txt 16 fuentes similares	< 1%		Palabras idénticas: < 1% (98 palabras)
2	repositorio.ucsg.edu.ec | Prevalencia de estrés laboral en el personal de salud d... http://repositorio.ucsg.edu.ec/bitstream/3317/21813/1/T-UCSG-POS-MSP-8.pdf 8 fuentes similares	< 1%		Palabras idénticas: < 1% (94 palabras)
3	repositorio.ucsg.edu.ec | Análisis y mejora de un sistema de gestión de inventari... http://repositorio.ucsg.edu.ec/bitstream/3317/22580/1/T-UCSG-PRE-ECO-ADM-720.pdf 8 fuentes similares	< 1%		Palabras idénticas: < 1% (94 palabras)
4	repositorio.ucsg.edu.ec | Inteligencia Artificial en la motivación de sentencia. http://repositorio.ucsg.edu.ec/bitstream/3317/23712/1/UCSG-C43-23243.pdf 7 fuentes similares	< 1%		Palabras idénticas: < 1% (84 palabras)
5	revistasinvestigacion.unmsm.edu.pe | Automatización del análisis exploratorio ... https://revistasinvestigacion.unmsm.edu.pe/index.php/ligeo/article/download/24493/19601	< 1%		Palabras idénticas: < 1% (57 palabras)

f. _____

Ing. Carrera Buri, Félix Miguel, Mgs.

AGRADECIMIENTO – MARÍA RENATA CHICA VEGA

Quiero agradecer en primer lugar a Dios por darme la vida, las fuerzas y la oportunidad de culminar mi etapa universitaria, gracias por cada reto porque sin duda eso me hizo más fuerte y responsable, sobre todo permitirme salir de mi zona de confort y tomar cada obstáculo como un aprendizaje de vida.

A mi madre, por ser mi ejemplo, por siempre enseñarme que las cosas se hacen con amor y a dar siempre lo mejor de mi sin esperar nada a cambio.

A mi padre, por ser mi mentor y siempre darme palabras y ánimos incluso cuando yo no los tenía, gracias por enseñarme a ser una persona servicial y confiar en mí.

A mis abuelos, mis tías, mi hermano y mi prima, que siempre me recibieron cada fin de semana con mucho amor y con ansias de verme llegar a casa.

A mi gatita Mavis, que adopté y se convirtió en mi compañera más fiel en mi último año universitario, la que se dormía en mi computadora y ronroneaba en mi pecho mientras hacía tesis.

Por supuesto agradecerle el Ing. Félix Carrera, quien tiene mi total admiración por el amor y paciencia en lo que hace y sobre todo por guiarme en este proceso importante, sin duda eso influyó positivamente en mi carrera universitaria.

Y, por último, pero no menos importante, a cada amigo verdadero que me hizo sentir como en casa en una ciudad que no es la mía.

DEDICATORIA – MARÍA RENATA CHICA VEGA

Dedico este logro a lo más importante que tengo en mi vida, a Dios, a mis padres, a mi hermano, a mis abuelos, a mis tías y mi prima, por haber estado orgullosos de mi desde el primer día, incluso cuando tuve que mudarme de ciudad con el fin de lograr una meta, siempre estuvieron apoyándome a la distancia y recibiéndome con los brazos abiertos. Agradezco a cada amigo/a que estuvo presente en estos 4 años. Y finalmente me lo dedico a mí misma, por todo el esfuerzo y responsabilidad que tuve durante el proceso, valió la pena cada noche sin dormir y es reconfortante saber que se hizo un gran trabajo.

AGRADECIMIENTO – MICHELA FRANCESCA SUCUZHANAY CERON

Hoy al terminar esta etapa tan importante de mi vida, miro hacia atrás y entiendo que este logro no es solo mío, sino también de todas las personas vitaminas que han estado para mí y han sido mi fuerza, mi motivación durante todo este camino.

Agradezco a Dios por sostenerme en cada momento, por darme serenidad cuando sentía incertidumbre y fortaleza cuando el cansancio parecía vencerme, en Dios siempre encontré la fuerza para continuar. Cada desafío fue una oportunidad para crecer, y hoy puedo decir que valió la pena.

A mis padres, Miguel y Jaqueline, gracias por ser mi apoyo incondicional. Gracias por creer en mí incluso cuando yo misma dudaba, por sus palabras de aliento, por sus consejos, por su paciencia y por cada sacrificio que hicieron para que pudiera cumplir esta meta. Todo lo que soy y todo lo que estoy logrando es y será gracias a ustedes y a los valores que me han enseñado. Ustedes han sido mi ejemplo de perseverancia y trabajo. Este título también es suyo.

A mi hermano Mattias, gracias por tu cariño tan sincero y por ser esa persona amorosa que siempre sabe cómo hacerme sentir mejor. En medio del estrés y las largas horas de trabajo, tu manera de apoyarme, tus palabras y tu presencia fueron un alivio. Tenerte como hermano es una de las mayores bendiciones de mi vida.

A mi tutor Félix, gracias por su el acompañamiento que nos ha dado a lo largo de este trabajo, por la ayudada brindada y por solventar cada duda que teníamos.

Finalmente, agradezco a todas las personas que, de una u otra forma, estuvieron presentes durante este proceso. Hoy cierro esta etapa con gratitud en el corazón y con la satisfacción de haber alcanzado una meta.

DEDICATORIA – MICHELA FRANCESCA SUCUZHANAY CERON

Me gustaría dedicarles este logro a las personas a quienes más amo y son pilares fundamentales en mi vida, a Dios, a mis padres, a mi hermano, a mis amigos que han estado para mí en las buenas y en las malas. Todo lo que hago es para ustedes, los amo.

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE ECONOMÍA Y EMPRESA
CARRERA DE NEGOCIOS INTERNACIONALES

TRIBUNAL DE SUSTENTACIÓN

f. _____

Ing. Carrera Buri, Félix Miguel, Mgs.

TUTOR

f. _____

Ing. Hurtado Cevallos, Gabriela Elizabeth, Mgs.

DIRECTORA DE CARRERA

f. _____

Lic. Freire Quintero César Enrique

COORDINADOR DEL ÁREA O DOCENTE DE LA CARRERA

INDICE

INDICE ILUSTRACIONES

Resumen

La presente investigación tuvo como objetivo implementar un modelo de *Machine Learning* para la clasificación y predicción de importación de repuestos tipo crítico en el sector automotriz. La problemática se fundamenta en la dependencia de métodos tradicionales de gestión de inventarios en pequeñas y medianas empresas, lo que genera ineficiencias en la cadena de suministro, sobrecostos operativos y dificultades para anticipar la demanda. Metodológicamente, se aplicó un modelo de aprendizaje no supervisado K-Means para clasificar los repuestos según su nivel de criticidad (alto, medio y bajo), considerando variables como rotación, impacto monetario, stock calculado e inconsistencia de inventario. Posteriormente, se desarrolló un modelo predictivo Random Forest Regressor para estimar la variable compras, evaluando su desempeño mediante el Error Absoluto Medio (MAE) y el coeficiente de determinación (R^2). Los resultados evidenciaron una alta concentración de repuestos clasificados como críticos y un desempeño predictivo óptimo del modelo, con un MAE de 1.18 unidades y un R^2 de 0.943, lo que indica una elevada capacidad explicativa. Se concluye que la aplicación de técnicas de aprendizaje automático mejora significativamente la gestión de inventarios, optimiza la planificación de importaciones y fortalece la competitividad empresarial en el sector automotriz.

Palabras clave: Machine Learning, gestión de inventarios, repuestos críticos, K-Means, Random Forest.

Abstract

This research aimed to implement a Machine Learning model for the classification and prediction of critical spare parts imports in the automotive sector. The problem is based on the reliance on traditional inventory management methods in small and medium-sized enterprises, which leads to supply chain inefficiencies, increased operational costs, and difficulties in demand forecasting. Methodologically, an unsupervised learning model, K-Means clustering, was applied to classify spare parts according to their level of criticality (high, medium, and low), considering variables such as turnover rate, monetary impact, calculated stock, and inventory inconsistency. Subsequently, a predictive model using Random Forest Regressor was developed to estimate the purchases variable, and its performance was evaluated using Mean Absolute Error (MAE) and the coefficient of determination (R^2). The results showed a high concentration of spare parts classified as critical and strong predictive performance, with an MAE of 1.18 units and an R^2 of 0.943, indicating high explanatory capacity. It is concluded that the application of Machine Learning techniques significantly improves inventory management, optimizes import planning, and strengthens business competitiveness in the automotive sector.

Keywords: Machine Learning, inventory management, critical spare parts, K-Means, Random Forest.

Résumé

Cette recherche avait pour objectif de mettre en œuvre un modèle de Machine Learning pour la classification et la prédiction des importations de pièces de rechange critiques dans le secteur automobile. La problématique repose sur la dépendance aux méthodes traditionnelles de gestion des stocks dans les petites et moyennes entreprises, ce qui entraîne des inefficacités dans la chaîne d'approvisionnement, une augmentation des coûts opérationnels et des difficultés dans la prévision de la demande. Méthodologiquement, un modèle d'apprentissage non supervisé, le clustering K-Means, a été appliqué afin de classifier les pièces selon leur niveau de criticité (élevé, moyen et faible), en tenant compte de variables telles que le taux de rotation, l'impact monétaire, le stock calculé et les incohérences d'inventaire. Par la suite, un modèle prédictif Random Forest Regressor a été développé pour estimer la variable achats, et sa performance a été évaluée à l'aide de l'erreur absolue moyenne (MAE) et du coefficient de détermination (R^2). Les résultats ont montré une forte concentration de pièces classées comme critiques ainsi qu'une excellente performance prédictive, avec un MAE de 1,18 unités et un R^2 de 0,943. Il est conclu que l'application des techniques de Machine Learning améliore significativement la gestion des stocks, optimise la planification des importations et renforce la compétitivité des entreprises du secteur automobile.

Mots-clés: Machine Learning, gestion des stocks, pièces critiques, K-Means, Random Forest.

Capítulo I: Generalidades de la Investigación

Introducción

Un elemento significativo para que los negocios crezcan a nivel mundial es el fuerte lazo que existe entre las compañías y lo nuevo en tecnología, las dos cosas han avanzado juntas y al mismo tiempo han crecido de manera exponencial, algo que ha funcionado muy bien con el tiempo. Esos giros cambiaron totalmente la economía del mundo, inventos grandes como la máquina de vapor durante la Revolución Industrial, el teléfono y la luz, que transformaron las formas de comunicación, y a partir del siglo XX apareció la automatización junto con las computadoras y el Internet en los noventa, esto inició una etapa nueva llamada Globalización que trajo mejoras serias al comercio y a la implementación sistemas digitales para manejar las empresas. Con el tiempo, las ciencias de la computación y la inteligencia artificial forjaron herramientas cruciales, suscitando un gran interés en la creación de una inteligencia similar a la humana en máquinas, volviéndose fundamental en la toma de decisiones empresariales y comprendiendo correctamente el funcionamiento del mundo comercial.

Uno de los diversos enfoques de la Inteligencia Artificial es la ciencia de datos, o Data Science. De acuerdo con lo planteado por Brodie, Michael L (2019):

> La ciencia de datos es un conjunto de principios y técnicas para aplicar métodos analíticos a datos a gran escala, incluyendo volumen, velocidad y variedad, con el fin de acelerar la investigación de los fenómenos representados por los datos, mediante la adquisición, preparación e integración de los mismos, posiblemente integrados con datos existentes, para descubrir correlaciones en los datos, con medidas de probabilidad y dentro de los límites de error.

Esta ciencia es aplicable en muchos sectores como en el gobierno, la educación, la salud, la banca, los negocios, la tecnología y el transporte.

Mediante la recolección de datos de estos sectores, la Inteligencia Artificial actúa como un sistema informático, clasificando y prediciendo mediante algoritmos basados en las tendencias y los comportamientos detectados en la información.

De hecho, existen algoritmos utilizados para la protección de los datos, mejorando la seguridad informática, el cifrado y la identificación personal como métodos que crean mecanismos firmes y aseguran la autenticidad y privacidad de la información. Dicha afirmación coincide con lo planteado por Iñiguez & Maldonado (2025):

> Los algoritmos de IA pueden analizar datos para crear perfiles detallados de individuos, que luego se utilizan en procesos como concesión de créditos, selección de personal o publicidades dirigidas. El perfilamiento automatizado es una de las aplicaciones más comunes de la IA y consiste en analizar datos para clasificar o predecir comportamientos, intereses y características de las personas (pág. 34).

La industria automotriz está fuertemente ligada con la IA, empezando por la creación de sistemas expertos como brazos robóticos, gemelos digitales, mantenimientos predictivos, y muchos otros sistemas implementados para el avance de esta industria, es por esto que el Machine Learning junto con la inteligencia de negocios son fundamentales en este sector industrial.

La convergencia de la inteligencia artificial en monitoreo en tiempo real, el almacenamiento de la carga, la demanda de las autopartes y el aprendizaje automático de los datos han mejorado la eficiencia operativa de las empresas.

Sin embargo, hoy en día en América Latina la adopción de estas tecnologías se ha implementado de forma fragmentada, generando poco impacto en su economía, lo contrario a otros países desarrollados. Tal argumento es respaldado por Poveda-Valverde (2025) que realizó 14 investigaciones acerca de las pymes en Latinoamérica, en sus hallazgos identificó 4 hechos en México, Brasil, Colombia y Argentina en el sector manufacturero, los cuales aplicaron las Inteligencia Artificial para la implementación de análisis predictivos, la automatización de la producción, la gestión de calidad, gestión de inventarios, y la identificación de errores. Por su parte Vergara Villegas et al. (2021) ofrecen una perspectiva más extensa sobre las tecnologías asociadas con la Industria 4.0, las cuales han crecido a nivel mundial. En Iberoamérica sus hallazgos reflejan que existen iniciativas preliminares basados en los 8 fundamentos que definen la Industria 4.0, esto incluye al internet de las cosas (Iot), cómputo en la nube, big data y analítica, ciberseguridad, robótica, realidad virtual y aumentada, fabricación aditiva e integración horizontal y vertical de sistemas. Pese a que muchos de estos proyectos aún están en una fase piloto, no deja de ser una realidad emergente en la región.

Rodríguez et al. (2021) plantean que “es fundamental implementar un sistema de gestión para evitar que el inventario se vea afectado por la variabilidad de la demanda”. Poder conocer qué producto tiene mayor salida en ventas y cual ya está obsoleto, para así poder mejorar la eficiencia y la rotación del inventario. Las empresas del sector automotriz enfrentan importantes desafíos como la gestión eficiente del inventario de repuestos. La necesidad de un sistema predictivo que ayude a determinar este tipo de factores, hará que se reduzcan los costos de la empresa y que se trabaje con mayor eficiencia. Un control ineficiente puede generar sobrecostos por exceso de stock. Ante este gran problema, surge la necesidad de implementar herramientas basadas en Machine

Learning que nos ayuden a determinar la demanda y así poder clasificarla según su nivel crítico.

Según Parra y Fuentes (2023), es importante resaltar que el objetivo de un sistema de gestión de inventario es mantener un registro de las compras, realizar un seguimiento de los bienes y suministros disponibles en stock, y gestionar la solicitud de estos cuando los niveles se reducen. Este sistema resulta fundamental para que las empresas puedan mantener una cadena de suministro organizada y sin contratiempos. Por ende, en el área automotriz, implementar el sistema de ML ha logrado optimizar la gestión de inventarios lo que ayudaría a la reducción de costos. En términos de adaptabilidad, los modelos de Machine Learning suelen ser más rápidos para adaptarse a nuevos entornos o tipos de datos. Esto se debe a su menor complejidad y a su capacidad para ser ajustados y afinados con menos datos y tiempo.

Por otro lado, Hernández et al. (2021) señalan que “el seguimiento de inventarios es una variable que tiene un impacto directo en la reducción de costos en las organizaciones”.

Es por esto que utilizar este algoritmo aumentaría la eficiencia de la empresa automotriz. Implementar soluciones basadas en IA disponibles ayuda a la empresa en el rendimiento, lo que podría pronosticar la demanda planificación empresarial integrada, optimización de planificación dinámica y automatización del flujo físico, todo lo cual se basa en modelos de predicción y análisis de correlación para comprender mejor las causas y efectos en las cadenas de suministro. Implementar con éxito la gestión de la cadena de suministro habilitada por IA puede permitir adoptar mejores costos logísticos.

El Machine Learning fundamenta su concepto en el desarrollo de algoritmos que facilitan a los sistemas informáticos a aprender a partir de experiencias pasadas, logrando

adaptarse de manera autónoma sin requerir intervención directa del ser humano. Los algoritmos de Machine Learning generalmente son menos complejos de interpretar, lo que resulta sencillo entender cómo generan un resultado. Esto resulta particularmente útil en ámbitos como la salud o las finanzas, dado que las decisiones automatizadas deben ser claras y comprensibles para las personas.

Existen situaciones en donde la información se transforma con frecuencia o provienen de distintas fuentes, los modelos de ML pueden ajustarse con menos esfuerzo y con menor costo computacional. A nivel micro, en países como Ecuador se dice que "está en un estado básico e incipiente en relación con el uso y aprovechamiento de la IA; así como una ausencia de política pública que fomente el aprovechamiento de la IA como medio de desarrollo social, económico y ambiental" (Barragán-Martínez, 2023, pág. 25).

Aunque la implementación de estas tecnologías aún es incipiente, hay esfuerzos por parte del sector público y privado para fomentar el uso de Machine Learning en el ámbito automotriz. Esto ha comenzado a abrir nuevas oportunidades para la modernización de sectores como la gestión de inventarios.

Esta alineación permite a las empresas abordar puntos clave de toma de decisiones con un nivel adecuado de información, mientras se evita la complejidad innecesaria.

Problemática

Actualmente una de las principales causas de los déficits de muchas pequeñas y medianas empresas (pymes) es la ausencia de la Inteligencia Artificial (IA), la resistencia de ser parte del cambio dentro de la competencia muchas veces se vuelve un desafío, las empresas aún son dependientes a un sistema tradicional de análisis que influye en la toma de decisiones, generando dudas sobre el uso de herramientas de la IA como el aprendizaje automático. Como menciona Jara Luis & Naspud Mayra (2025):

> Las pequeñas y medianas empresas (PYMEs) de Ecuador tienen la oportunidad de utilizar y aprovechar esta tecnología, conocida como Inteligencia Artificial, sin embargo, se enfrentan a obstáculos considerables en su implementación. Uno de los desafíos más grandes es la careza de conocimiento y la poca experiencia en IA por parte de las PYMEs.

En consecuencia, si no se aplica Machine Learning para el control de repuestos críticos en la industria automotriz, resulta evidente que generaría un sin número de impactos desfavorables para un crecimiento óptimo y eficiente.

Según los estudios de Feizabadi (2022) haciendo una comparativa entre los modelos predictivos y los modelos tradicionales, encuentra que “la falta de modelos predictivos para prever la demanda en la cadena de suministro podría generar un funcionamiento lento de los procesos de una empresa y decisiones menos eficaces, por lo tanto, un inventario ineficiente con altos costos operativos”.

En el entorno corporativo, la evidencia de un buen desarrollo empresarial se gestiona mediante decisiones, mismas que son determinadas por un análisis de datos, información extraída con la finalidad de estudiar sus hallazgos y patrones para optimizar

procesos, los cuales recopilan grandes cantidades de datos, es por esta razón que es por esta razón que, sin modelos predictivos y de clasificación basados en ML, las medidas de importación de las autopartes implicaría basarse únicamente un sistema manual y antiguo o en hechos históricos de la empresa que ralentiza el crecimiento y la competitividad. Esto conlleva que, se limite la capacidad de analizar datos operativos, técnicos, logísticos, comerciales, el registro de las ventas, el tipo de repuesto, las fluctuaciones de la demanda, disponibilidad del inventario, etc.

"Los métodos tradicionales de previsión suelen tener dificultades para considerar las condiciones dinámicas del mercado, lo que genera ineficiencias y disrupciones" (Kagalwala, Radhakrishnan, Mohammed, Kothinti, & Kulkarni, 2025, pág. 142).

En este caso, el riesgo aumenta cuando se genera un desabastecimiento de las existencias, o por su parte un excedente de las mismas, puesto que al no disponer de patrones que alimenten algoritmos es complejo predecir la demanda o cuales son repuestos críticos. No tener un suficiente abasto de inventario genera oportunidades para que otras empresas se beneficien de la escasez de recursos de sus competidores, en consecuencia, los clientes al experimentar la escasez de piezas o retrasos optan por otras marcas y las empresas pierden la fidelidad de sus compradores y pierden potencia en el mercado competitivo.

Considerando lo anteriormente mencionado, las organizaciones se enfrentan a otro tipo de circunstancias, como problemas en el área logística, dificultad con los proveedores e incertidumbre con la demanda.

Además, la clasificación de los repuestos tipo crítico es esencial, debido a que si no se priorizan adecuadamente no habría una buena segmentación, una minimización de fallos y una gestión óptima de las autopartes según su nivel de criticidad, en este sentido

las organizaciones sobreestiman su límite de inventario e invierten en repuestos de poco impacto, a diferencia de aquellos faltantes que son realmente críticos, generando un desbalance financiero significativo al no saber con precisión que y cuantas autopartes se necesitan.

Hoy en día la falta de implementación del aprendizaje automático va de la mano con una mala toma de decisiones, puesto que esta “es fundamental en este contexto, ya que cualquier error o retraso puede tener repercusiones significativas en la eficiencia y la rentabilidad de toda la cadena de suministro” (Calle García, Pincay Delgado, Mendoza Pionce, & Bravo Quijije, 2024, pág. 268)

Es decir que esto podría generar que una cadena de suministros se transforme en una menos ágil, con mayor presupuesto, generando más errores y con una menor capacidad de solidez en el mercado, afectando directamente a la rentabilidad del mantenimiento y productividad de las empresas.

No obstante, sin modelos predictivos que identifiquen los que son críticos, las empresas dependen únicamente de registros históricos para determinar cuáles son aquellos que regularmente presentan fallas, provocando una baja productividad y tiempo inactivos de automóviles, sin anticipar la probabilidad del tiempo de falla a través de patrones asociados con la temperatura, el uso o vibración. Al no predecir las necesidades de los consumidores se podrían generar quiebres en el stock en temporadas claves.

En conclusión, la falta de sistemas predictivos y de clasificación para la importación de repuestos tipo crítico limita a las empresas a generar adaptabilidad en un sector industrial automatizado y competitivo. De igual manera, el condicionamiento de seguir implementando herramientas convencionales trae consigo un retraso financiero, organizacional y tecnológico.

Justificación

La Inteligencia Artificial, o IA, que ha ganado terreno desde sus inicios en el siglo veinte, tiene como fin imitar lo que el ser humano puede hacer en labores difíciles usando aparatos que funcionan solos. (Russell & Norvig, 2021). En esta área, sobresale el Aprendizaje Automático, o ML, el cual se centra en lograr que los ordenadores hagan mejor su trabajo al mostrarles mucha información, si bien existen distinciones importantes en cómo manejan y qué tan elaborados son los cálculos. (Bishop, 2006).

Los avances tecnológicos que se han desarrollado en la actualidad son impulsados por herramientas como el Machine Learning (ML). Este trabajo tiene como objetivo clasificar y predecir los repuestos para así saber cuál tiene mayor acogida y así poder importarlo. Entender cómo funciona el Machine Learning nos ayuda a entender las posibles soluciones que nos pueden dar y así poder sacarle el mayor provecho en áreas en donde este ha tenido un crecimiento rápido en los últimos años.

En el ámbito de la gestión de inventarios, diversos estudios han señalado que la optimización basada en datos permite mejorar los niveles de servicio y disminuir costos operativos. La correcta identificación de productos críticos facilita una mejor planificación del stock de seguridad y una reducción del capital inmovilizado (Chopra & Meindl, 2019). En este sentido, integrar modelos de Machine Learning en la toma de decisiones logísticas no solo mejora la eficiencia operativa, sino que también fortalece la sostenibilidad financiera de las empresas del sector automotriz.

Además, el uso de técnicas predictivas en cadenas de suministro ha demostrado impactos positivos en la reducción de incertidumbre y en la mejora del desempeño organizacional. La analítica avanzada permite anticipar fluctuaciones en la demanda y responder de manera más ágil a cambios del mercado (Fawcett & Provost, 2013). Para

empresas dedicadas a la importación de repuestos, esto representa una herramienta estratégica que disminuye riesgos asociados a la sobrecompra o al desabastecimiento de productos de alta rotación.

La tecnología de Aprendizaje Automático emplea reglas matemáticas para detectar tendencias dentro de la información, lo cual ha facilitado hacer tareas solas y optimizar las elecciones tomadas en muchas industrias, probando ser útil para vaticinar y examinar información, siendo mejor si hay poca información o si las tendencias son sencillas de descubrir.

La incorporación de modelos de aprendizaje automático en procesos empresariales responde a una tendencia global hacia la transformación digital. La capacidad de los algoritmos para aprender de datos históricos y mejorar su desempeño con el tiempo convierte a estas herramientas en instrumentos valiosos para la innovación y la competitividad (Goodfellow et al., 2016).

Este Aprendizaje Mecánico se convirtió en algo esencial al desarrollar mecanismos aptos para procesar mucha data y entregar resultados futuros, transformando totalmente la manera en que las empresas conciben sus estrategias operativas. Gracias a esto, las organizaciones logran resguardar sus bienes y hacer que el cliente se sienta mejor con servicios hechos justo a la medida de lo que requieren.

La gestión eficiente del inventario es un aspecto crítico para garantizar la disponibilidad de productos y evitar pérdidas económicas derivadas de faltantes o “stockouts”. Un estudio publicado en Journal of Digital Economy destaca que, en contextos de alta incertidumbre y gran volumen de datos, los enfoques que incorporan múltiples variables relevantes del inventario y patrones de ventas recientes pueden mejorar de manera significativa la predicción de faltantes, en comparación con modelos

tradicionales que dependen únicamente de proyecciones de largo plazo (Liu et al., 2025). Los autores encontraron que factores como los niveles actuales de inventario, pronósticos de demanda a corto plazo y datos de ventas recientes tienen mayor poder predictivo para anticipar rupturas de stock, lo cual es especialmente relevante para negocios con miles de unidades de artículos (como repuestos automotrices). Estos hallazgos subrayan la importancia de utilizar indicadores de corto plazo para mejorar la precisión de las predicciones y apoyar decisiones de reposición más oportunas, fortaleciendo así la eficiencia operativa general de la cadena de suministro.

Este proyecto también busca identificar los repuestos críticos de manera eficaz y con los mínimos errores. El Machine Learning tiene algoritmos más sencillos y requiere menos capacidad de cálculo, lo que lo hace adecuado para empresas con recursos limitados, proporcionando una guía para saber que repuesto está en el mercado, comprenderlo y saber mediante la predicción y el algoritmo tecnologías y decidir cuándo utilizar cada una. En el mundo automotriz, aplicar estos sistemas puede ofrecer una ventaja al facilitar la innovación y la adaptación rápida a los cambios del mercado.

El motivo por el cual usamos el aprendizaje automático en este estudio se basa en que puede bajar la duda y los gastos ligados a traer y guardar piezas de reemplazo fundamentales. Mediante esta tecnología, los programas informáticos logran agrupar las partes según cuánto se piden, cuánto duran, qué tan a menudo se rompen y qué tan rápido salen del almacén. Así, las compañías del rubro de autos obtienen un panorama completo que les deja prever faltantes, eludir tener mucha existencia y asegurar que las partes clave siempre estén ahí, mejorando la forma de trabajar y contentando más a quienes nos compran.

Este estudio tiene sentido porque ofrece algo valioso al progreso técnico y saber del país, al juntar saberes de la inteligencia artificial, el examen de información y el manejo de rutas de entrega en un único esquema práctico. Así, aparte de ayudar al progreso en la academia, también refuerza la mejora en la industria, empujando a las compañías de autos a tener un cambio digital más firme, productivo y con más capacidad de competir.

En conclusión, este estudio se justifica por la importancia de conocer cómo la IA con su rama el Machine Learning aportan distintas soluciones, en este caso para identificar el repuesto según su criticidad. Tener una comprensión nítida de estas herramientas tecnológicas facilita elegir sabiamente cómo usarlas, mejorando mucho lo que se consigue e impulsando el progreso en variados campos. Además, esto ayuda a ser más imaginativo al enfrentar retos y destapa caminos inéditos para crear cosas nuevas en diversas esferas.

Alcance

El presente trabajo de titulación con el tema *Aplicación del Machine Learning para la clasificación y predicción e importación de repuestos tipo crítico para automotores* tiene como público objetivo profesionales como los analistas de datos o aquellos especializados en logística y gestión de inventarios que busquen información actualizada del Machine Learning y la importancia de implementar y enseñar estas herramientas en el mundo corporativo para la interpretación de los hallazgos.

Para los profesionales en Operaciones y Supply Chain les resulta fundamental aplicar sistemas que van de la mano con la inteligencia de negocios, puesto que son útiles para una planificación adecuada del inventario, y más importante aún, para tomar decisiones en base a la información de los datos. Además, facilita a las organizaciones a optimizar la estructura y la gestión de su cadena de suministro mediante la reducción de costos y la mitigación de riesgos (Govindan Kannan, 2018). De esta manera, pueden tener un histórico de datos, clasificar los bienes, predecir la demanda de estos para posteriormente importar el stock requerido y así obtener un sistema óptimo en el inventario, todo gracias al conocimiento y uso del aprendizaje automático.

Asimismo, los profesionales que manejan el campo de la ciencia de datos, los cuales son prácticos y competitivos, es por esto que se mantienen informados ante los avances tecnológicos y dedican tiempo al procesamiento de los datos para una correcta interpretación. Herramientas como revistas, artículos científicos, libros, informes de consultoría y ensayos académicos son de gran ayuda para el dominio del aprendizaje automático y la anticipación del mercado.

Objetivos

Objetivo general

- Implementar un modelo de Machine Learning para la clasificación y predicción de importación de repuestos tipo crítico para automotores.

Objetivos específicos

1. Revisar la literatura científica sobre la aplicación del Machine Learning, con el fin de ver su enfoque, algoritmos y modelos aplicados que respalden su aplicación en la clasificación y predicción de repuestos críticos.
2. Desarrollar y aplicar un modelo de clasificación y predicción eficiente para la importación de repuestos tipo crítico.
3. Discutir los hallazgos en donde se evidencia el algoritmo aplicado e interpretar los resultados similares obtenidos que comparen el uso de la metodología y su aporte para la optimización de las autopartes en el sector automotriz.

Capítulo II: Fundamentos Teóricos

Marco teórico

Inteligencia Artificial

La Inteligencia Artificial, conocida como la IA, es un campo muy extenso de conocimientos computacionales que van de la mano con un conjunto de datos. “La IA basada en el conocimiento, que empezó a desarrollarse a finales de los años setenta, intenta modelar el conocimiento humano mediante modelos informáticos” (López de Mántaras & Brunet, 2023, pág. 13).

En palabras de Caiafa y Lew (2020) se destaca los siguiente:

> Se define a la IA como la capacidad de las máquinas de tomar decisiones efectivas, es decir que, minimizan la probabilidad de cometer errores en cada decisión tomada. Además, se busca diseñar máquinas que puedan “aprender” y adquirir conocimiento a partir de experiencias o patrones según su contexto, tal como lo hacen los humanos y otras especies animales, quienes han desarrollado la habilidad de decidir basándose en la experiencia.

Por otro lado, Lasse Rouhiainen (2018) definió a la inteligencia artificial como “la capacidad de las máquinas para ejecutar tareas que normalmente requieren inteligencia humana” (p.17). En otras palabras, son procesos que por lo general lo haría el ser humano, sin embargo, la IA se ha convertido en un gran aliado.

Aprendizaje Automático

El aprendizaje automático, conocido como Machine learning (ML) es la disciplina que permite programar computadoras para que aprendan a partir de datos, esta rama de la IA se ha desarrollado gracias a la programación y grandes volúmenes de datos generados por las empresas, con el propósito de almacenar, entrenar, evaluar y analizar la información para la automatización de procesos y una correcta toma de decisiones. Con el transcurso del tiempo, ha tenido un impacto significativo, marcando un antes y un después en ámbitos sociales, económicos y de sanidad. Citando a Bobadilla (2020) define al Machine Learning como "la ciencia que logra que los ordenadores "aprenden" a partir de información…el campo del Machine Learning está destinado al desarrollo de modelos que pueden extraer patrones de diferentes tipos de datos mediante el uso de algoritmos" (pág. 13).

Existen diferentes tipos de aprendizajes, entre los más comunes se encuentran: el aprendizaje supervisado y el aprendizaje no supervisado.

Aprendizaje supervisado

"Los algoritmos de aprendizaje supervisado funcionan a partir de datos etiquetados, intentando encontrar una función de hipótesis que, al recibir las variables de entrada, asigne la etiqueta de salida correspondiente" (Valencia, 2025, pág. 41). Es decir, se aplica cuando cada dato, o conjunto de datos de entrada (muestra) son previamente etiquetados.

De acuerdo con Contretas & Padilla (2024) mencionan que:

> Se le llama supervisado porque conforme el modelo predice las salidas para datos de prueba, se calcula el error entre lo predicho por el algoritmo y el valor real. El

objetivo es minimizar el error, ajustando la función de densidad de probabilidad que conecta las entradas con las salidas.

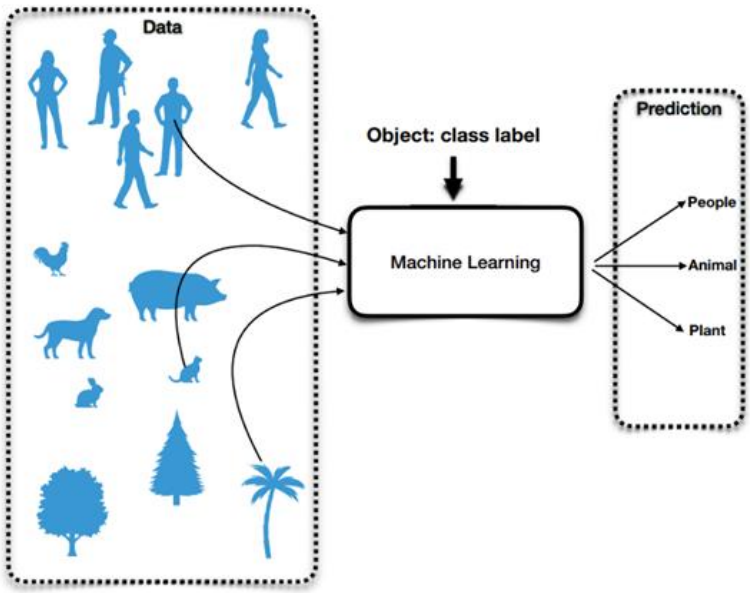
Clasificación

Según lo establecido por Verma & Gupta (2018) “Clasificar es el mecanismo de minería de datos aplicado con mayor frecuencia, que toma un conjunto de ejemplos preclasificados para desarrollar un modelo que puede clasificar una población (conjunto de datos) de registros en general” (pág. 48). De acuerdo con Contretas & Padilla (2024) mencionan que:

> Los datos se encuentran etiquetados, lo que significa que se asigna una clase o categoría a la variable dependiente, por ejemplo, "vive" o "no vive", "aprueba" o "no aprueba". El objetivo en este tipo de tarea es que el modelo pueda asignar una etiqueta a un nuevo dato no visto previamente por el modelo, por ejemplo, determinar si "aprueba" o "no aprueba" (pág. 17)

La clasificación se diferencia de otros métodos de Machine Learning puesto que trabaja con datos etiquetados. En este sentido, la clasificación es diferente a la regresión, que predice valores numéricos (precio, temperatura, etc.) y asimismo es diferente del clustering, ya que este pertenece el aprendizaje no supervisado que agrupa los datos según la similitud de los datos no etiquetados.

Figura 1 Método Supervisado - Clasificación



Nota. Método Supervisado - Clasificación. [imagen] por Font (2019)
https://openaccess.uoc.edu/server/api/core/bitstreams/25c73efb-0646-4811-83f1-aa1eb3772896/content

Regresión

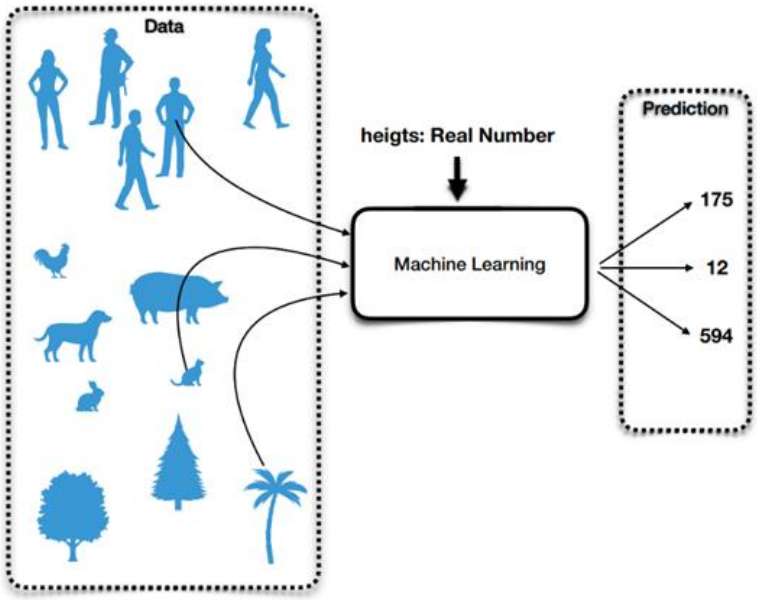
En contraste con la clasificación, en los problemas de regresión se busca predecir un valor numérico en lugar de un valor etiquetado. Los datos se etiquetan con un valor real o flotante. “La regresión lineal permite predecir el comportamiento de una variable (dependiente o predicha) a partir de otra (independiente o predictora). Tiene presunciones como la linealidad de la relación, la normalidad, la aleatoriedad de la muestra y homogeneidad de varianzas.” (Dagnino S., 2024).

Como indica Lizares (2017, citado en Aco Tito, Hancco Condori & Pérez Vera, 2023) la Regresión:

> Es una técnica estadística de varias variables, que facilita el análisis de la relación que se encuentra entre un grupo de variables independientes y una variable dependiente, esta última es dicotómica, es decir que solo puede tomar dos valores,

que generalmente son cero y uno, donde cero es ausencia y uno presencia (pág. 87).

Figura 2 Método Supervisado – Regresión



Nota. Método Supervisado - Regresión. [imagen] por Font (2019) https://openaccess.uoc.edu/server/api/core/bitstreams/25c73efb-0646-4811-83f1-aa1eb3772896/conten

Mínimos cuadrados

De acuerdo con Moral (2016):

El método de los mínimos cuadrados, consiste en calcular la suma de las distancias al cuadrado, estas distancias van entre los puntos reales y los puntos definidos por la recta estimada a partir de las variables introducidas en el modelo, de esta forma la mejor estimación será aquella que minimice estas distancias (pág. 202).

Regresión Lineal Múltiple

La Regresión Lineal Múltiple (RLM) es un algoritmo utilizado "cuando se busca predecir el valor de una variable medida en una escala continua, de razones o intervalos,

en función del valor de otras dos o más variables. En este caso, la variable que se pretende predecir es la variable dependiente" (Carlos, Manuel, Eduardo, 2023, pág. 4).

Supuestos del modelo de regresión múltiple:

Para la construcción de un modelo de regresión lineal es necesario que se cumpla con los siguientes supuestos estadísticos (Vilà Baños, Torrado Fonseca, & Reguant Álvarez, 2019) entre ellos:

- Linealidad: Que la relación entre las variables sea lineal.
- Independencia: Que los errores en la medición de las variables explicativas sean independientes entre sí.
- Homocedasticidad: Que los errores tienen varianza constante.
- Normalidad: Que las variables sigan la Ley Normal.
- No colinealidad: Que las variables independientes no estén correlacionadas entre ellas.

Aprendizaje No supervisado

Este tipo de aprendizaje utiliza información no etiquetada, es decir que consta una base de datos de entradas más no de variables de salida, basándose en la ausencia de cualquier supervisor y, por lo tanto, de medidas de error absolutas.

De acuerdo con Cazares y Pico (2025):

El algoritmo principal de este tipo es k medias, que permite agrupar o segmentar observaciones en grupos basándose en características y distancias entre cada una de las observaciones. Esta agrupación consiste en realizar una minimización de la

suma de las distancias entre cada uno de los objetos y el centroide de su grupo (clúster).

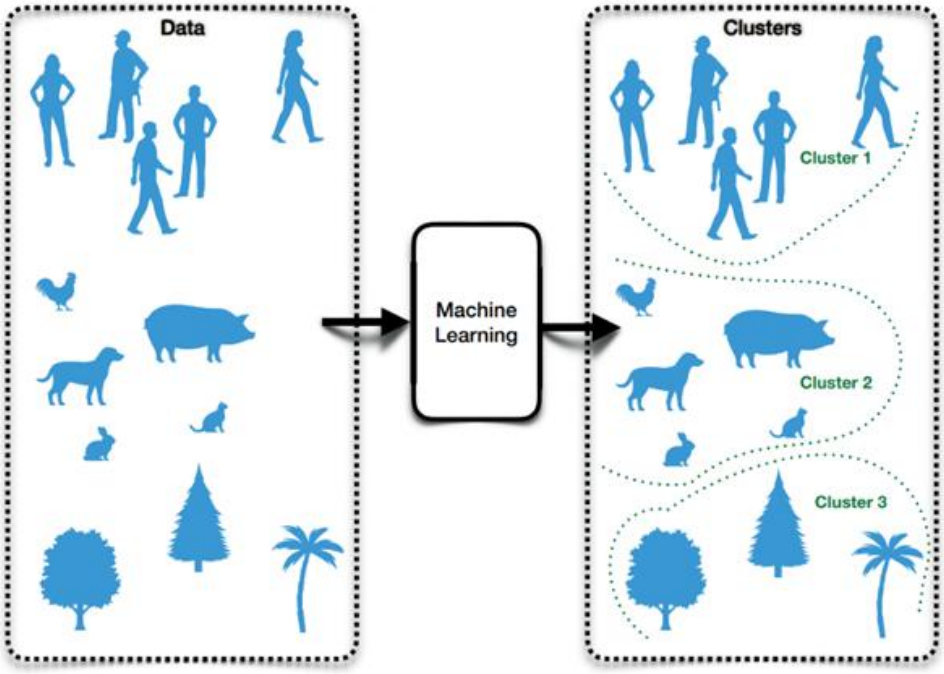
Es decir que los algoritmos no supervisados se encargan por sí solos de descubrir y presentar estructuras de datos. El conjunto de datos es la recolección de información no etiquetada $D = \{x_i\}$ $N_i=1$, donde *x* es un vector de características, y el objetivo del algoritmo del aprendizaje no supervisado es crear un modelo a partir del vector de características x, siendo esta la variable de entrada y posteriormente transformarlo en otro vector, intentado encontrar patrones en los datos.

También es conocido como *descubrimiento de conocimientos*, ya que no menciona qué tipo de patrones se deben buscar y no existe una métrica de error que pueda utilizarse, en contraste con el aprendizaje supervisado, en donde se puede comparar la predicción de *y* para un *x* dado con el valor observado).

Clustering

Una de las técnicas más utilizadas para categorizar datos es el método clúster, que permite agrupar observaciones semejantes de un conjunto definido. Siguiendo a Font (2019) "al mismo tiempo, se pretende que observaciones pertenecientes a grupos diferentes tengan poca similitud. En otras palabras, se busca maximizar la distancia entre los grupos" (pág. 10).

Figura 3 Método No Supervisado



Nota. Método No Supervisado - clustering. [imagen] por Font (2019)
https://openaccess.uoc.edu/server/api/core/bitstreams/25c73efb-0646-4811-83f1-aa1eb3772896/content

K-means

K-means es un algoritmo de aprendizaje no supervisado empleado para agrupar un conjunto de observaciones en K clusters, de manera que los elementos de un mismo cluster presentan mayor similitud posible entre sí. En palabras de Xu y Tian (2015) lo describen como "un método de partición que busca minimizar la suma de distancias entre cada punto y el centroide de su clúster, mediante un proceso que ajusta iterativamente la asignación de datos y la posición de los centroides hasta alcanzar la convergencia del algoritmo".

Por su parte, Jain (2010) destaca que el algoritmo "K-means es una de las técnicas de clustering más utilizadas principalmente porque es fácil de comprender, tiene eficiencia computacional y puede aplicarse en distintos contextos, como en la segmentación de clientes, análisis de patrones y gestión de inventarios".

En el contexto de repuestos en el área automotriz, el algoritmo K-means puede emplearse para agrupar repuestos según el comportamiento de demanda, la frecuencia de uso, tiempos de entrega, su tipo de criticidad y su nivel de criticidad. Este clasificador permite facilitar el diseño de políticas diferenciadas de reposición e importación.

Sin embargo, tanto Jain (2010) como Xu y Tian (2015) advierten que "los resultados de K-means pueden variar según la elección del número de clústeres y de la inicialización de los centroides. Por esta razón, es recomendable utilizar métricas de validación y entrenar el algoritmo varias veces con distintas inicializaciones para obtener mejores resultados".

A pesar de estas limitaciones, su rapidez y facilidad de implementación hacen que K-means sea muy útil como herramienta de apoyo para la clasificación preliminar de repuestos antes de aplicar otros métodos más complejos.

Árbol de decisión

Los árboles de decisiones o decision tree se han convertido en uno de los algoritmos más utilizados dentro del aprendizaje automático, principalmente porque su forma de trabajar y su forma jerárquica trata que los procesos de toma de decisiones resulten fáciles de interpretar.

De acuerdo con un artículo publicado en la *Revista Varianza* de la Universidad Mayor de San Andrés (2021):

> Un árbol de decisión es un modelo empleado tanto como para tareas de clasificación como de regresión. Este algoritmo se construye mediante una estructura tipo árbol, compuesta por un nodo interno que representa una prueba

sobre una característica, las ramas reflejan los resultados de esa prueba y los nodos finales representan una decisión final o clase asignada.

Según Rokach y Maimon (2008):

> Un árbol de decisión es un modelo predictivo que organiza el espacio de los datos en grupos homogéneos mediante reglas de decisión aplicadas en cada nodo interno, hasta llegar a nodos finales (nodos hoja) que representan una clase o un valor de salida.

Por otro lado, Oday F (2022) señala que:

> Los árboles de decisión son modelos ampliamente utilizados para la clasificación y la regresión en el aprendizaje automatizado o también denominado Machine Learning, principalmente cuando se requiere trabajar con grandes cantidades de datos estadísticos, tal y como sucede hoy es la Ciencia de Datos y el Big Data.

Asimismo, estos autores destacan que los árboles de decisión son especialmente útiles en situaciones donde la interpretabilidad es un factor clave según el giro del negocio, porque permiten identificar aquellas variables que influyen en cada decisión (Rokach y Maimon, 2008). Esto resulta importante en la gestión de los repuestos y su tipo de criticidad, ya que es posible establecer reglas como, por ejemplo: si el tiempo de importación es elevado y la frecuencia de falla es alta, por lo tanto, el repuesto se clasifica como crítico.

Por otro lado, Biau y Scornet (2016) señalan que:

> Los árboles de decisión pueden utilizarse tanto de forma independiente o formando parte de métodos de ensamblaje, gracias a su flexibilidad para trabajar

con variables categóricas y numéricas, así como su capacidad de captar ciertas relaciones no lineales en las variables de entrada y la variable objetivo.

No obstante, los autores también advierten que un solo árbol puede ser inestable y débil antes pequeñas variaciones en los datos, por lo que suele complementarse con técnicas más robustas como Random Forest, que combinan múltiples árboles para generar mejores resultados.

Sobreajuste

Según Jason Brownlee (2020) "El sobreajuste ocurre cuando un modelo de aprendizaje automático aprende demasiado bien aquellos patrones encontrados del conjunto de entrenamiento, incluso captan el ruido, valores atípicos o variaciones aleatorias que no representan al problema real" La consecuencia de esto es que el modelo podría fallar durante el entrenamiento cuando se evalúa con datos nuevos.

Brownlee explica que el sobreajuste se presenta cuando el modelo se vuelve innecesariamente complejo en relación a la cantidad de datos disponibles, inicialmente generando un desempeño aparentemente bueno durante el entrenamiento, pero un resultado pobre en la etapa de la validación. Esto lo convierte en uno de los problemas más comunes en modelos como los árboles de decisión, ya que tienden a crecer sin restricciones y memorizar los datos.

Poda

Según Blockeel et al. (2023), "la poda es un proceso que reduce la complejidad de un árbol de decisión eliminando ramas que no contribuyen de forma significativa al rendimiento del modelo".

Los autores explican que la poda permite evitar el sobreajuste, ya que los árboles tienden a crecer demasiado y memorizar el ruido del conjunto de entrenamiento. La poda, en consecuencia, mejora la capacidad del modelo para generalizar, al simplificar la estructura del árbol sin comprometer su precisión. Además, detallan que las técnicas modernas de poda forman parte de estrategias de aprendizaje responsable, donde se prioriza la interpretabilidad y la estabilidad del modelo frente a datos nuevos.

Cross Validation

La validación cruzada es una de las métricas más utilizadas para evaluar la capacidad de un modelo de generalización. Arlot y Celisse (2010) describen la validación cruzada como "el proceso que divide los datos en subconjuntos que sirven para entrenar y probar el modelo repetidamente, permitiendo estimar su error real y detectar sobreajuste".

Vector

Gersho y Gray (2012) describen a un vector como "un conjunto ordenado de características que representan completamente un punto dentro de un espacio de múltiples dimensiones" (p. 4).

En el contexto del aprendizaje automático, un vector representa a cada instancia o registro de un conjunto de datos mediante un vector que agrupa valores cuantificables. Esta representación vectorial permite que algoritmos como Random Forest procesen y comparen observaciones en términos matemáticos, facilitando análisis complejos como clasificación y predicción.

Nodo

En los árboles de decisión que componen Random Forest, un nodo es el punto donde se realiza una evaluación o prueba sobre una característica del conjunto de datos. Según Rokach y Maimon (2008), un nodo es "la unidad fundamental de un árbol de decisión donde se aplica un test sobre un atributo para dividir el conjunto de instancias" (p. 23). Los nodos internos contienen reglas de decisión, mientras que los nodos hoja representan el resultado final del proceso de clasificación.

Impureza

La impureza es una medida del nivel de desorden o mezcla de clases dentro de un nodo de un árbol. Lim, Loh y Shih (2013) explican que la impureza "indica el grado de heterogeneidad dentro de un nodo y determina la calidad de una división" (p. 12). Cuanto más homogéneo sea un nodo, menor será su nivel de impureza. Por esta razón, los algoritmos intentan dividir los datos de forma que se reduzca esta medida. La impureza es esencial para calcular métricas fundamentales como el índice Gini o la entropía para la calidad de cada división.

Split

Una división (split) es el paso mediante el cual los datos tienden a separarse en ramas más homogéneas dentro de un árbol de decisión. Breiman, Friedman, Olshen y Stone (2017) explican que este proceso consiste en "la división de un nodo en subgrupos similares a partir de una regla basada en un solo atributo" (p. 104).

Debido a estas divisiones, esta operación permite que los árboles de decisión generen estructuras con rutas lógicas y claras que faciliten la clasificación de las

instancias según sus características, haciendo que el proceso de decisión sea comprensible.

Bagging

Según Kuhn y Johnson (2013), “bagging es una técnica de ensamblaje basada en entrenar varios modelos del mismo tipo utilizando diferentes muestras del conjunto de datos obtenidas mediante remuestreo bootstrap”.

Los autores explican que, al combinar las predicciones de estos modelos, generalmente mediante el promedio, se logra reducir la varianza y aumentar la estabilidad del algoritmo. De esta manera el modelo se optimiza a partir de una versión mejorada distinta de los datos. Este enfoque resulta útil en modelos inestables como los árboles de decisión. En este sentido, el bagging representa el fundamento matemático del algoritmo Random Forest, puesto que permite crear árboles diversos generando un modelo más robusto ante ruido y variabilidad de los datos.

Bootstrap

La técnica de bootstrap es un método estadístico de remuestreo que genera varios subconjuntos del conjunto original al momento de seleccionar observaciones de manera aleatoria y con reemplazo.

Según Efron y Tibshirani (1994), indican que “el bootstrap facilita la estimación de la variabilidad del modelo y la creación de distribuciones de error sin requerir de supuestos paramétricos estrictos”. En el caso de Random Forest, cada árbol se entrena sobre una muestra bootstrap única, lo que introduce una distinción entre los árboles del bosque, mejorando la estabilidad y la robustez del modelo final.

Out-of-Bag

Cuando se habla del error Out-of-Bag (OOB) se hace referencia a una estimación interna del rendimiento predictivo en modelos hechos por técnicas como el bagging y Random Forest. En palabras de Hastie, Tibshirani y Friedman (2009):

> En cada muestra bootstrap empleada para el entrenamiento de un árbol, cerca del 37% de las observaciones no son incluidas en dicha muestra. Estas observaciones que no han sido vistas en la fase del entrenamiento, son utilizadas como datos de validación para el árbol. Al juntar los resultados generados por los árboles, se puede calcular el error del modelo sin realizar la separación de los datos en conjuntos de entrenamiento y prueba.

En este sentido, el Out-of-Bag es un método eficiente para evaluar el error y la capacidad de generalización de Random Forest sin perder la precisión y la correcta ejecución de los datos.

Random Forest

De acuerdo con Jason Brownlee (2022), “Random Forest es parte de los algoritmos del aprendizaje supervisado, el cual genera múltiples árboles de decisión, es decir un bosque aleatorio que combina sus predicciones para obtener un modelo más robusto y preciso”.

El autor explica que, a diferencia de un árbol individual el cual puede ser inestable se vuelve propenso al sobreajuste, es por esto que el Random Forest utiliza el principio de “sabiduría de conjunto” o *ensemble learning*, donde muchos modelos débiles se integran para producir un modelo más robusto. Esto permite que el algoritmo mantenga una alta precisión incluso en datasets complejas o con ruido.

De acuerdo con Towards Data Science, Sharma (2021) explica que “el bagging funciona creando múltiples modelos entrenados sobre diferentes muestras bootstrap del mismo dataset, reduciendo la varianza del modelo y aumentando la precisión en comparación con un solo árbol”. El autor destaca que el bagging es un mecanismo clave para que Random Forest logre estabilidad.

Por otro lado, Biau y Scornet (2016) describen Random Forest como un método de ensamblaje que genera múltiples árboles construidos con muestreo *bootstrap* y selección aleatoria de variables en cada división, y cuya predicción final se obtiene promediando (en regresión) o votando (en clasificación) las salidas de todos los árboles.

De acuerdo con Belgiu y Drăguţ (2016), Random Forest se ha consolidado como uno de los algoritmos más robustos y precisos para problemas de clasificación y regresión, en parte porque es poco sensible al sobreajuste y puede manejar conjuntos de datos con alta dimensionalidad y correlación entre variables.

Random Forest permite calcular la importancia de las variables. Biau y Scornet (2016) explican que el algoritmo mide cuánto empeora el rendimiento del modelo cuando se altera aleatoriamente una variable específica; de este modo se puede saber qué factores son más influyentes en la predicción. En la clasificación de repuestos tipo crítico, esto ayuda a justificar, con base en datos, si la criticidad depende más del tiempo de importación, de la disponibilidad de proveedores o del impacto económico del paro de máquina.

Métricas de evaluación de modelos

Matriz de confusión

Una matriz de confusión es una métrica utilizada para evaluar el desempeño de los clasificadores binarios, la cual consta de una estructura 2 x 2, basada en observaciones representadas como VP, VN, FP, FN. A través de esta representación, es posible evaluar de manera clara el comportamiento del modelo. Tal y como afirma Rainio, Teuho & Klén (2024):

> Cada etiqueta binaria predicha tiene cuatro posibles resultados: un verdadero positivo (VP) ocurre cuando el resultado positivo predicho está correcto, mientras que un verdadero negativo (VN) corresponde a un resultado negativo bien predicho, un falso positivo (FP) es una instancia negativa predicha como positiva, y un falso negativo (FN) es una instancia positiva predicha como negativa

Accuracy

También conocida como *exactitud*, "esta medida se utiliza para evaluar qué tan bien el modelo clasifica correctamente todas las clases, tanto las positivas como las negativas." (Mendoza, Macías, Morales, & Cedeño, 2024) Es decir que se refiere a la proporción de las predicciones correctas con respecto al total de resultados positivos y negativos.

Precisión

La precisión también puede interpretarse como el grado de confianza de las predicciones positivas. En otros términos, "esta medida permite evaluar la calidad de un modelo de clasificación en machine learning, principalmente utilizada para evaluar qué

tan bien el modelo identifica correctamente los verdaderos positivos entre todos los positivos predichos.” (Mendoza, Macías, Morales, & Cedeño, 2024)

Recall

El Recall o comúnmente conocido como sensibilidad, se enfoca en la capacidad de detectar los valores positivos entre todos los positivos. “Es la métrica de evaluación de modelos de clasificación que mide la proporción de casos positivos que son identificados correctamente por el modelo”. En otras palabras, el Recall mide la capacidad del modelo para encontrar todos los casos positivos. (Mendoza, Macías, Morales, & Cedeño, 2024)

F1-score

El F1-score surge como una métrica que genera un equilibrio, se utiliza para integrar el Recall y la Precisión del modelo y es la media armónica de ambos. “Combina la precisión y el Recall. La precisión indica la proporción de predicciones positivas que son correctas, mientras que el Recall mide la proporción de casos positivos que se identifican correctamente por el modelo.” (Mendoza, Macías, Morales, & Cedeño, 2024)

Es por esto que, esta métrica es útil cuando se busca un desempeño balanceado, sin presentar debilidades en los indicadores.

MAE

Esta métrica permite identificar la distancia entre los valores predichos con respecto a los valores reales. Es decir que, **“**mide el promedio de las diferencias absolutas entre el valor estimado del modelo y los valores reales.” (Terven, Cordova Esparza, Romero González, Pedraza, & Chávez-Urbiola, 2025). Ofrece un nivel del error en la predicción.

RMSE

El RMSE o Raíz del Error Cuadrático Medio es una métrica de evaluación utilizada principalmente en el desempeño del análisis de regresión. Según lo planteado por Soto & González (2019):

> El RMSE mide, en promedio, cuánto se alejan los datos observados de los estimados mediante la aplicación de la regresión. Si el modelo de regresión estimado tiene intercepto 0, pendiente 1,0 (lo que se ha llamado la línea 1:1 o identidad) y el RMSE es 0, entonces se tiene el ajuste ideal y, por tanto, la precisión para simular el modelo, es perfecto (pág. 521).

ROC

La curva ROC o también conocida como Receiver Operating Characteristics, es una herramienta que permite evaluar de forma visual, organizar y seleccionar modelos en función a su eficacia y hacer una comparativa del desempeño del modelo. Textualmente, los autores Martínez & Pérez (2023) señalan que:

> La curva ROC es una herramienta estadística que se utiliza para evaluar la capacidad discriminativa de una prueba diagnóstica dicotómica. Estas curvas representan la sensibilidad en función de los falsos positivos, es decir, el complementario de la especificidad para distintos puntos de corte (pág. 1).

Esto facilita una comprensión clara de los aciertos y errores del modelo. En la curva de ROC se encuentra un área llamada área bajo la curva (AUC), la cual evalúa la facultad discriminativa del modelo.

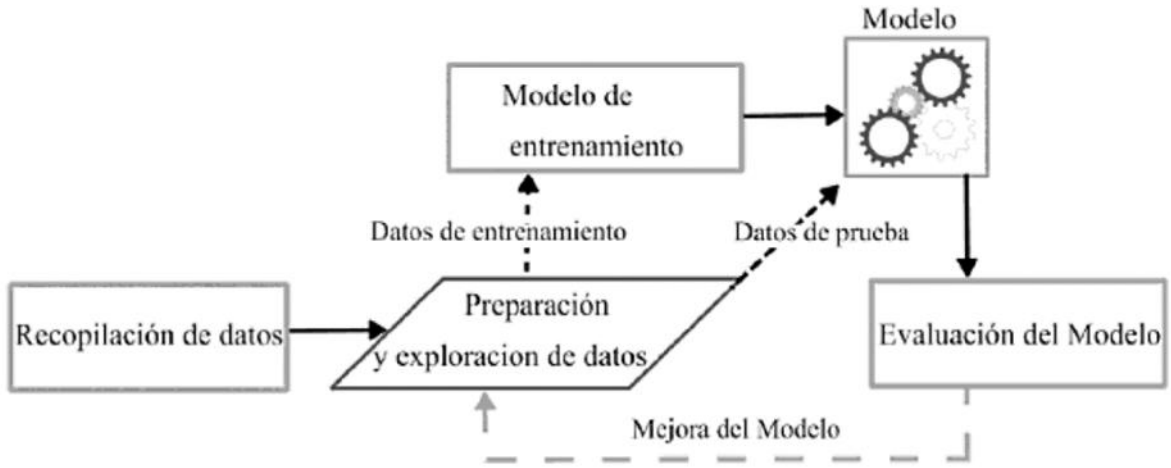
AUC

El AUC o también conocida como Area Under the Curve:

"Es utilizada para evaluar la eficacia de los modelos de clasificación, principalmente en conjuntos de datos desequilibrados. midiendo su capacidad para diferenciar entre clases positivas y negativas del conjunto de datos, sin depender del umbral de decisión que se utiliza para las predicciones." (Mendoza, Macías, Morales, & Cedeño, 2024)

Etapas de la implementación del aprendizaje de máquina

Figura 4 Fases del proceso de aplicación del Machine Learning



Nota. Etapas de implementación de un algoritmo de aprendizaje de maquina [imagen] por Cuevas et. al (2021) https://books.google.es/books?hl=es&lr=&id=Ek1OEAAAQBAJ&oi=fnd&pg=PT15&dq=que+es+el+machine+learning&ots=6EBFEreWyl&sig=0aTk2E5x1PABbC1txIpwuVbPm44#v=onepage&q&f=false

Para comprender correctamente la implementación y el desarrollo de un algoritmo de aprendizaje automático, es fundamental reconocer cuáles son las distintas fases que conforman a un modelo. Este proceso parte de la recopilación de datos y avanza hasta la evaluación del modelo, con el objetivo de obtener respuestas efectivas y aptas para el problema a tratar.

1. **Recopilación de datos:** esta primera etapa consiste en obtener información/datos que sirvan de guía para que el algoritmo pueda operar mediante una base de datos y genere resultados.
2. **Preparación y exploración de datos:** para que un aprendizaje automatizado funcione de manera exitosa es crucial que trabaje en la preparación de la calidad de los datos, lo que significa realizar una limpieza de datos, ya sean datos desordenados, duplicados, información innecesaria, datos mal categorizados o aquellos que requieren algún cambio.
3. **Entrenamiento del modelo:** una vez que se realizó el data cleaning, los datos son utilizados para alimentar al algoritmo seleccionado, con la finalidad de aprender patrones y predicciones a partir del entrenamiento generando así un modelo.
4. **Evaluación del modelo:** es importante realizar una evaluación del rendimiento del modelo, puesto que cada reacción diferente y pueden generar ciertos errores sistemáticos al aprender datos. Esta evaluación permite determinar qué tan eficiente es el aprendizaje durante el entrenamiento, dependiendo del modelo a emplear es posible evaluar qué tan preciso es mediante datos de prueba o indicadores de rendimiento según el caso.
5. **Mejora del modelo:** en caso de que los resultados no sean satisfactorios, existe la posibilidad de utilizar herramientas para mejorar el rendimiento del modelo, en algunos casos se necesita cambiar a un algoritmo que se ajuste al problema, o a su vez agregar más datos representativos al entrenamiento.

Definición de inventario

La administración del inventario es fundamental para mantener las finanzas de las empresas en orden, de la mano con la productividad, puesto que tienen poca liquidez al ser un activo corriente, generando mayor rentabilidad. Bajo esta perspectiva, en palabras de Durán (2012):

> Los inventarios son el centro de la comercialización de las empresas puesto que comprenden los bienes o productos para la compra-venta o la producción de los mismos con el fin de comercializarlos, es decir que, son parte del funcionamiento de la producción para gestionar la demanda (pág. 56).

Es por esta razón que, la clasificación de las autopartes en el inventario permite definir aquellos repuestos de mayor a menor criticidad basándose en una base de datos en términos de representación de compra y venta.

Faltantes (stockouts)

Considerando la definición y la importancia del inventario, existe también un concepto que define aquellos faltantes, conocida también como Stockouts, "es la situación de desabastecimiento cuando la demanda de un artículo en inventario supera la oferta y no se puede satisfacer la necesidad de dicho artículo" (Sánchez Ruiz, Blanco, & Kyguoliené, 2018).

Repuestos Tipo Crítico

Los repuestos críticos son aquellos componentes cuya ausencia afecta directamente la continuidad operativa, la seguridad o el rendimiento de un sistema técnico o industrial. Según Cavalieri et al. (2008), "un repuesto se considera crítico cuando su

falla genera tiempos muertos significativos, costos elevados o riesgos operativos que comprometen la disponibilidad del equipo".

Los autores enfatizan que la criticidad depende tanto del impacto de la falla como del tiempo requerido para reponer el componente, especialmente en sistemas donde la importación implica largos plazos logísticos. Los repuestos críticos presentan características que los diferencian de otros componentes.

De acuerdo con Ghodrati y Kumar (2005), "estos repuestos suelen tener alta importancia funcional, baja disponibilidad en el mercado, tiempos prolongados de adquisición y un efecto directo sobre la detención del sistema".

Señalan que estos componentes requieren estrategias de administración específicas debido a la dificultad para prever su demanda y por los costos elevados asociados a su falta.

Diversos factores influyen en que un repuesto sea clasificado como crítico. Según los hallazgos de García et al. (2017) identifican que "la criticidad puede evaluarse en función de variables como: impacto en la seguridad del usuario, consecuencias económicas de la falla, probabilidad de ocurrencia, tiempo de reposición (lead time), disponibilidad de proveedores y necesidad operativa". La combinación de estos factores permite priorizar los repuestos que requieren mayor control en inventarios o importación.

Marco conceptual

Random Forest Regression

De acuerdo con lo planteado por hahboun & Maarouf (2021):

El bosque aleatorio (RF) es un método de regresión basado en conjuntos que tiene estructura de árbol, el cual muestra las relaciones entre las características y el objetivo, lo que permite obtener resultados fáciles de entender e interpretar. Este método es una adaptación del árbol de decisión, en el que un modelo produce predicciones basadas en una sucesión de modelos base (pág. 3).

$$g(x) = f1(x) + f2(x) + f3(x) + \cdots + fk(x)$$

En este caso, cada modelo base es un árbol de decisión.

g(x): modelo final

x: conjunto de variables de entrada.

$f_1(x) + f_2(x) + \cdots + f_k(x)$: modelos base

k: número total de árboles de decisión.

Regresión lineal

Los modelos de regresión son principalmente utilizados para predecir un resultado (a menudo notado como Y) en función de diversas variables explicativas (representadas como $X_1, X_2, \ldots, X_n$) (Font, 2019, pág. 24). Es decir que, a partir de un conjunto de variables explicativas, es posible estimar la variable de estudio (variable objetivo).

El modelo lineal simple se expresa mediante:

$$Y_i = \beta_0 + \beta_1 X_i + \ldots + \beta_n X_n + \varepsilon_i$$

En donde:

Y_i : variable dependiente en la observación *i*

X_i : variable independiente

β_0 : intercepto

β_1 : pendiente o coeficiente de regresión

ε_i : término de error aleatorio

Por un lado, la ecuación de la regresión está compuesta por una parte determinística que representa la relación entre la variable dependiente (respuesta) y las variables independientes (explicativas), por otro lado, está la parte no determinística o aleatoria representada por ε que corresponde al error.

Mínimos Cuadrados Ordinarios

El método de regresión que minimiza la suma de los cuadrados de los residuos, se basa en el criterio para el ajuste de las curvas, reduciendo la diferencia entre los puntos y la curva.

$$min_{\beta 0,\beta 1} \sum_{i=1}^{n} (Y_1 - \hat{Y}_i)^2$$

Sus soluciones analíticas son las siguientes

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n} (X_i - \bar{X})(Y_i - \bar{Y})}{\sum_{i=1}^{n} (X_i - \bar{X})^2}$$

$$\hat{\beta}_0 = \bar{Y} - \hat{\beta}_1 \bar{X}$$

En donde las $\bar{X}$ y las $\bar{Y}$ son las medias.

Modelo de regresión lineal múltiple

Para *k* variables explicativas, el modelo es:

$$Y_i = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + \cdots + \beta_k X_i + \varepsilon_i$$

Su forma matricial es la siguiente:

$$Y = X\beta + \varepsilon$$

En donde:

Y: vector n×1 de la variable dependiente

X: matriz n× (k+1) de variables independientes (incluye columna de unos).

β: vector de coeficientes.

ε: vector de errores.

K-means

Paso de Asignación de datos mediante la siguiente ecuación:

$$\underset{ci \in C}{arg\,min}\, dist\,(c_{i,}\, x)^2$$

En donde:

x : un dato

c_i : un centroide

Paso de actualización del centroide

C: conjunto de todos los centroides

$dist\ (c_{i,}\ x)^2$: distancia al cuadrado entre el dato y el centroide

$arg\ \ min$: elige el que minimiza.

$$c_i = \frac{1}{|S_i|}\sum_{xi\ \in Si} x_i$$

En donde:

S_i *:* conjunto de puntos asignados al cluster *i*

$/S_i/$*:* número de puntos en ese cluster

$\sum$ *xi*: suma de todos los puntos del cluster

El centroide nuevo es el promedio de los puntos del grupo.

Para esto el K-means minimiza la suma de las distancias cuadradas de cada grupo mediante la siguiente ecuación:

$$\sum_{i=1}^{k}\sum_{x\in s1}\|x - c_1\|^2$$

Estimación matricial por OLS

$$\widehat{\beta}\ = (x^t x)^{-1} x^t y$$

Predicción

$$\hat{y} = X\hat{\beta}$$

Residuos

$$e = y - \hat{y}$$

Métricas de evaluación

Matriz de confusion

La matriz de confusión resume el desempeño del modelo mostrando los aciertos y errores.

	$Pred.Positiva$	$Pred.Negativa$
$Real\ Positiva$	VP	FN
$Real\ Negativa$	FP	VN

Donde:

$$TP = verdaderos\ positivos$$

$$TN = verdaderos\ negativos$$

$$FP = falsos\ positivos$$

$$FN = falsos\ negativos$$

F1-Score

Mientras que las anteriores métricas de evaluación se hacen directamente sobre las predicciones realizadas, el f1 score no es más que la media armónica entre precisión y recall, es decir mientras más altos sean los valores de dichas métricas, mayor será el resultado el F1 Score, si uno de ambos valores es bajo, el valor en sí mismo del F1 Score se verá afectado drásticamente. El F1 Score se mide en una escala de 1 a 0, o de manera porcentual y es el resultado de la media armónica entre el Recall y la Precision, consecuentemente presenta una mayor utilidad a la hora de interpretar el balance que tiene el modelo de redes neuronales en cuanto a su capacidad para poder detectar positivos y que efectivamente dichos hallazgos sean correctos.

$$F1 = \frac{2 \times Precisión \times Recall}{Precisión + Recall}$$

Exactitud (Accuracy)

El Accuracy nos sirve como métrica de evaluación para visualizar la proporción de predicciones correctas sobre el total de observaciones, mide que tanto el modelo va predice de manera correcta el outcome. Es decir, mide que tanto evalúa de manera correcta el modelo.

En casos de redes neuronales perceptrón el Accuracy puede ser llegar a ser perjudicialmente engañosa si alguna de las clases presenta una mayor frecuencia que las otras, es decir; es de mayor utilidad cuando las clases son balanceadas. El Accuracy se mide en una escala de 1 a 0, o de manera porcentual.

$$Accuracy = \frac{VP + VN}{VP + VN + FP + FN}$$

Donde:

$$VP = verdaderos\ positivos$$

$$VN = verdaderos\ negativos$$

$$FP = falsos\ positivos$$

$$FN = falsos\ negativos$$

Se calcula de la división donde en el numerador se encuentran el total de predicciones correctas, ya sea positiva o negativa, mientras que en el denominador se encuentran todas las predicciones realizadas.

Precision

La precisión mide en una escala del 0 a 1, o porcentualmente que tanto fueron realizadas las predicciones positivas de manera correcta, al medir la proporción de cuantos casos clasificados como positivos son realmente positivos.

Mientras más alto sea el valor de precisión podemos interpretar que el modelo de redes neuronales comete menos falsos positivos, es decir; clasificar erróneamente un negativo como positivo.

$$Precisión = \frac{VP}{VP + FP}$$

Donde:

$$VP = verdaderos\ positivos$$

$$FP = falsos\ positivos$$

Se calcula de la división donde en el numerador se encuentran el total de predicciones positivas verdaderas, mientras que en el denominador se encuentran todas las predicciones positivas realizadas (falsas y verdaderas).

Recall

El Recall nos sirve como métrica de medición para visualizar que tanto el modelo de redes neuronales predice de manera correcta las observaciones positivas reales. Se podría interpretar como la capacidad del modelo para detectar cualquier tipo de observaciones positivas, minimizando de esta manera los falsos negativos.

$$Recall = \frac{VP}{VP + FN}$$

Donde:

$$TP = verdaderos\ positivos$$

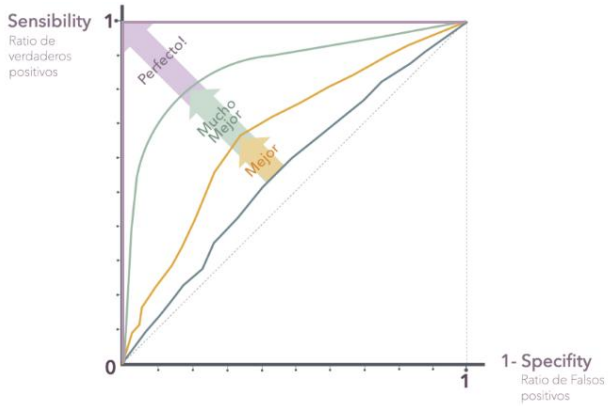
$$FN = falsos\ negativos$$

El Recall se mide en una escala de 1 a 0, o de manera porcentual mediante la división donde del total de predicciones positivas verdaderas, la suma de las predicciones positivas verdaderas y las predicciones negativas falsas.

Curva Roc

Se trata de un gráfico en dos dimensiones en el que el eje horizontal muestra la tasa de falsos positivos, mientras que el eje vertical representa la tasa de verdaderos positivos.

Figura 5 Curva de Sensibilidad vs. Especificidad



La idea es partir de un conjunto pequeño de observaciones con:

1. **Una variable binaria**

$$\gamma \in \{0,1\} \text{ y}$$

2. **Una probabilidad predicha p^∈ [0,1].**

$$\hat{p} \in \{0,1\}$$

3. **Clasificamos como positivo si**

$$\hat{p} \geq t$$

Variamos el *umbral* t desde +∞ hasta −∞. Para cada t calculamos:

$$\text{Sensibilidad} = \frac{TP}{P}$$

$$Especificidad = \frac{TN}{N}$$

$$J = Sensibilidad + Especificidad - 1$$

La curva ROC es la siguiente:

$$\text{ROC} = \{(\text{FPR}(\tau), \text{TPR}(\tau)) : \tau \in \mathbb{R}\}$$

La línea diagonal que une los puntos (0,0) y (1,1) indica el desempeño de un clasificador aleatorio. Un modelo efectivo debe ubicarse por encima de dicha diagonal y aproximarse al punto óptimo (0,1), el cual representa una clasificación ideal, sin presencia de falsos positivos ni falsos negativos.

Marco Legal

El presente estudio utiliza información proporcionada por una corporación vinculada al sector camaronero. En función de ello, el desarrollo de la investigación se realiza siguiendo los lineamientos de la Superintendencia de Protección de Datos. Si bien el proyecto no implica el tratamiento de datos personales, ya que se trabaja con información de carácter productivo y operativo, sí se emplean datos privados de la empresa. Por tal motivo, se considera necesario definir el siguiente marco legal y ético para su tratamiento.

En este contexto, se toma como referencia la Ley Orgánica de Protección de Datos Personales (2021), que constituye la norma nacional aplicable. En particular, su Artículo 2 excluye de su ámbito de aplicación a los datos anonimizados o a aquellos que no permitan identificar a una persona natural. No obstante, los principios recogidos en el Artículo 10 de dicha Ley se asumen como guía para la gestión de la información empleada en esta investigación, debido a su relevancia para la gobernanza, seguridad y uso responsable de los datos.

a) Juridicidad

El manejo de la información facilitada por la empresa se efectuará dentro de un marco de legalidad y respeto a la normativa vigente, garantizando que el uso de los datos sea legítimo y acorde con el carácter científico-investigativo del proyecto (Ley Orgánica de Protección de Datos Personales, 2021, art. 10).

b) Lealtad

La información será tratada de manera honesta y respetuosa hacia la empresa que la provee, evitando cualquier práctica que pueda ocasionar perjuicios,

interpretaciones erróneas o usos encubiertos distintos a los informados (Ley Orgánica de Protección de Datos Personales, 2021, art. 10).

c) Transparencia

La empresa colaboradora contará con información clara sobre el objetivo del tratamiento, las fases del proceso y el alcance del uso de los datos durante el desarrollo del modelo, asegurando visibilidad y comprensión en cada etapa del proyecto (Ley Orgánica de Protección de Datos Personales, 2021, art. 10).

d) Finalidad

Los datos se utilizarán exclusivamente para el diseño, entrenamiento y validación del modelo, orientado a la optimización de los procesos productivos en el cultivo de camarón, sin destinarlos a actividades ajenas a los fines de la investigación (Ley Orgánica de Protección de Datos Personales, 2021, art. 10).

e) Pertinencia y minimización

Únicamente se solicitará la información estrictamente indispensable para el funcionamiento del modelo, evitando incorporar variables que no aporten valor analítico ni capacidad predictiva al estudio (Ley Orgánica de Protección de Datos Personales, 2021, art. 10).

f) Proporcionalidad

El volumen y tipo de datos a emplear se ajustará al alcance del proyecto, evitando requerir o procesar información que exceda las necesidades reales

del desarrollo de la investigación (Ley Orgánica de Protección de Datos Personales, 2021, art. 10).

g) Confidencialidad

La información proporcionada será tratada con carácter reservado y se protegerá frente a accesos o divulgaciones no autorizadas, reconociendo su importancia estratégica para la empresa (Ley Orgánica de Protección de Datos Personales, 2021, art. 10).

h) Calidad y exactitud

Se trabajará con datos verificados, depurados y actualizados, considerando que la precisión de la información es clave para obtener resultados confiables y reducir el margen de error del modelo (Ley Orgánica de Protección de Datos Personales, 2021, art. 10).

i) Seguridad de los datos

Se implementarán medidas técnicas y administrativas que disminuyan los riesgos de pérdida, modificación o acceso indebido a la información, especialmente al emplear herramientas y plataformas de carácter digital (Ley Orgánica de Protección de Datos Personales, 2021, art. 10).

j) Responsabilidad proactiva

Se registrarán y documentarán los procedimientos vinculados al tratamiento de datos, con el fin de asegurar trazabilidad, control y la posibilidad de demostrar la aplicación de buenas prácticas en su gestión (Ley Orgánica de Protección de Datos Personales, 2021, art. 10).

k) Aplicación favorable al titular

Ante cualquier situación no prevista dentro de este proceso, se optará por decisiones que prioricen la protección y el beneficio de la empresa que suministra la información, manteniendo siempre un enfoque preventivo en el manejo de los datos (Ley Orgánica de Protección de Datos Personales, 2021, art. 10).

Metodología

El objetivo del presente capitulo es exponer y desarrollar los métodos que se utilizaron en la aplicación de un modelo de clasificación para estimar la criticidad de los repuestos de la empresa 4M, en este caso se aplicó K-means, algoritmo del aprendizaje no supervisado. Posteriormente se aplicó árbol de clasificación para determinar el grado de criticidad y finalmente se realizó la predicción para estimar la nueva cantidad a exportarse. El proyecto se realizará mediante una investigación con enfoque cuantitativo, a partir de la base de datos obtenida de la empresa de repuestos de automóviles 4 M.

Lenguaje de programación para análisis avanzado de datos: Python

Python es un lenguaje de programación sencillo diseñado para que sea fácil de aprender y de usar. En el artículo *El lenguaje de programación Python* (2014) se menciona que "fue creado por Guido van Rossum, un programador holandés a finales de los 80 y principio de los 90 cuando se encontraba trabajando en el sistema operativo Amoeba"

"Python es de propósito general y produce código fácilmente legible, comprensible para quienes no son programadores. Además, gracias a su extensibilidad intrínseca, existe una gran cantidad de bibliotecas y módulos de terceros" (Hosmer, 2014).

En otras palabras, Python es uno de los múltiples lenguajes de programación, reconocido por su versatilidad y su sintaxis clara y legible.

"Python también permite la programación imperativa, programación funcional y programación orientada a aspectos" (González Duque, pág. 8).Está apto para la programación orientada a objetos, funcional e imperativa y tiene una variedad de librerías y generalmente utilizado para el desarrollo de aplicaciones web, software, aprendizaje

automático y la ciencia de datos. Su uso es sencillo puesto que funciona en cualquier sistema operativo sin generar cambios significativos.

Tipo de investigación

El método de investigación que se ha implementado en el proyecto para la clasificación y predicción de repuestos críticos según el tipo de datos recolectados para realizar la investigación es de tipo cuantitativo por aprendizaje automatizado de los datos.

Para poder desarrollar el presente estudio se siguieron 5 procesos:

Proceso 1: Data Clean

Paso 1: Análisis exploratorio de datos

Pandas

La librería Pandas fue utilizada para la lectura, manipulación y estructuración de los datos provenientes de archivos en formato Excel. Mediante la estructura denominada DataFrame, permitió organizar la información en tablas bidimensionales, facilitando procesos como la limpieza de valores nulos, la transformación de variables y la selección de columnas relevantes para el análisis. Pandas es ampliamente utilizada en ciencia de datos por su eficiencia en el manejo de datos estructurados y su integración con bibliotecas numéricas y de aprendizaje automático (McKinney, 2010).

El alias pd se asigna con el fin de simplificar la escritura del código y mejorar su legibilidad dentro del entorno Google Colab.

NumPy

La librería NumPy fue empleada como soporte fundamental para el manejo de arreglos numéricos y la ejecución de operaciones matemáticas sobre el conjunto de datos.

Su estructura basada en arrays multidimensionales permite realizar cálculos de manera eficiente y optimizada, lo que resulta esencial en procesos de análisis estadístico y en la implementación de algoritmos de aprendizaje automático. NumPy constituye uno de los pilares del ecosistema científico en Python debido a su rendimiento y versatilidad (Harris et al., 2020).

Matplotlib

La librería de Matplotlib se especializa en el análisis, visualización y modelamiento de grandes conjuntos de datos (Lemenkova 2020)

Se utilizó para la visualización gráfica de los datos y de los resultados obtenidos por el modelo. Esta herramienta permite generar gráficos que facilitan la interpretación del comportamiento de las variables y la evaluación del desempeño del algoritmo aplicado.

El submódulo pyplot, importado con el alias plt, facilita la creación y personalización de gráficos dentro del entorno Google Colab.

Seaborn

Seaborn genera gráficos estadísticos de alto nivel que se integra directamente con la librería de Pandas (Waskom 2021)

Librería de visualización estadística basada en Matplotlib, utilizada para realizar análisis exploratorios de los datos. Permite generar gráficos más estéticos y descriptivos, facilitando la identificación de patrones, relaciones y distribuciones entre las variables analizadas.

En este proyecto, Seaborn fue empleada para complementar el análisis gráfico de los datos previo a la aplicación del modelo predictivo.

Paso 2: Cargar los datos desde el equipo local

Para cargar la base de datos utilizada en el presente trabajo, se empleó la funcionalidad de carga de archivos desde el equipo local que ofrece el entorno Google Colab. Esta opción permite importar archivos directamente desde la computadora del investigador hacia el entorno de ejecución en la nube, facilitando el acceso a la información sin necesidad de conexiones externas a bases de datos.

Para ello, se utilizó el siguiente código:

```
from google.colab import files

uploaded = files.upload()
```

- **from google.colab import files**: importa el módulo files, el cual forma parte de las herramientas propias de Google Colab y permite gestionar la carga y descarga de archivos.
- **files.upload()**: abre una ventana interactiva que permite seleccionar uno o varios archivos desde el equipo local y cargarlos al entorno de trabajo.
- **uploaded**: es una variable que almacena temporalmente los archivos cargados, permitiendo su posterior lectura y procesamiento dentro del cuaderno de Google Colab.

Este procedimiento fue utilizado para importar el archivo en formato Excel que contiene la base de datos de repuestos, la cual posteriormente fue procesada mediante la librería Pandas para su análisis y preparación antes de la ejecución de los modelos de Machine Learning.

Paso 3: Lectura de datos desde un archivo Excel

Una vez cargado el archivo desde el equipo local al entorno de **Google Colab**, se procedió a la lectura de la base de datos en formato Excel utilizando la librería **Pandas**, la cual permite trabajar con archivos estructurados de manera eficiente.

Para ello, se empleó el siguiente código:

data = pd.read_excel('Repuestos_automotores')

- **pd.read_excel()**: es una función de la librería Pandas que permite leer archivos en formato Excel y convertirlos en una estructura tipo *DataFrame*.
- **'Repuestos_automotores'**: corresponde al nombre del archivo que contiene la base de datos con la información de los repuestos.
- **data**: es el *DataFrame* que almacena los datos cargados, organizados en filas y columnas, facilitando su posterior análisis y procesamiento.

Mediante esta instrucción, la información contenida en el archivo Excel es importada correctamente al entorno de Google Colab, permitiendo su visualización, exploración y transformación previa a la aplicación de los modelos de clasificación y predicción utilizados en la investigación.

Paso 4: Limpieza de datos

La limpieza de datos constituye una etapa fundamental dentro del proceso de análisis y modelado en proyectos de ciencia de datos y aprendizaje automático. Esta fase tiene como objetivo garantizar la calidad, consistencia y confiabilidad de la información utilizada, ya que la presencia de datos incompletos, inconsistentes o erróneos puede afectar significativamente el desempeño y la precisión de los modelos predictivos.

En el contexto de la presente investigación, la base de datos utilizada proviene de registros reales de una empresa de repuestos automotrices, lo cual implica la posible

existencia de valores faltantes, inconsistencias en el inventario, errores de registro o formatos no homogéneos. Por esta razón, fue necesario aplicar un proceso de limpieza de datos previo a la implementación de los algoritmos de clasificación y predicción.

Paso 5: Corrección de nombres de columnas

Durante el proceso de limpieza de datos, se identificaron inconsistencias en los nombres de algunas columnas, producto de errores de escritura en la base de datos original. Este tipo de inconsistencias puede generar problemas durante la ejecución del modelo, ya que los algoritmos de aprendizaje automático requieren que los nombres de las variables sean exactos y coherentes a lo largo de todo el proceso.

Para corregir este inconveniente, se aplicó el siguiente código:

```
data.rename(columns={"Stock Calculado": "Stock_Calculado"}, inplace=True)
```

- **data.rename()**: es una función de la librería Pandas que permite modificar los nombres de las columnas de un *DataFrame*.
- **columns={"Stock Calcualdo": "Stock_Calculado"}**: define un diccionario que indica el nombre incorrecto de la columna y su corrección correspondiente.
- **inplace=True**: especifica que el cambio se realiza directamente sobre el DataFrame original, sin crear una copia adicional.

Paso 6: Función para convertir strings con comas a float

Durante la revisión del conjunto de datos, se identificó que algunas variables numéricas se encontraban almacenadas en formato de texto (string), debido a la presencia de separadores de miles (comas) o valores no reconocibles como números. Esta situación es común en bases de datos provenientes de sistemas administrativos o registros manuales

y puede generar errores en los modelos de aprendizaje automático si no se corrige adecuadamente.

Definición de la función de limpieza numérica:

def clean_numeric(x):

- **def**: es una palabra reservada del lenguaje Python utilizada para definir una función.
- **clean_numeric**: es el nombre asignado a la función, el cual describe su propósito, que es la limpieza y conversión de valores numéricos.
- **(x)**: representa el parámetro de entrada de la función, es decir, el valor individual que será evaluado y procesado. Este valor puede corresponder a un dato de una columna del conjunto de datos.

Esta línea marca el inicio de una función personalizada diseñada para recibir un valor a la vez y aplicar sobre él un proceso de validación, limpieza y conversión a formato numérico. La creación de esta función permite reutilizar el mismo procedimiento de limpieza sobre múltiples variables del conjunto de datos, garantizando consistencia y eficiencia en el preprocesamiento de la información.

```
if pd.isna(x):
```

La función *pd.isna(x)* se utiliza para identificar valores faltantes dentro del conjunto de datos. Su propósito es detectar aquellos registros que no contienen información válida, comúnmente representados como valores nulos (*NaN*). Al reconocer estos casos, se evita aplicar transformaciones o conversiones sobre datos inexistentes, lo cual podría generar errores o distorsionar los resultad.

```
return np.nan
```

El uso de *np.nan* permite representar explícitamente la ausencia de un valor numérico dentro del conjunto de datos. Al devolver este valor, se conserva la información como un dato faltante, evitando asignaciones incorrectas o conversiones forzadas que puedan alterar la calidad de los datos.

```
if isinstance(x, str):
```

La función *isinstance(x, str)* se utiliza para identificar si un valor se encuentra almacenado en formato de texto. En bases de datos reales, es común que ciertos valores numéricos sean registrados como cadenas de caracteres debido a formatos de entrada, separadores de miles u otros símbolos que impiden su reconocimiento inmediato como datos numéricos.

```
x = x.replace(',', '')
```

La instrucción x.replace(',', '') se emplea para eliminar las comas que aparecen en valores numéricos registrados como texto. En bases de datos provenientes de sistemas administrativos o registros manuales, es frecuente que las comas se utilicen como separadores de miles, lo cual impide que dichos valores sean interpretados correctamente como datos numéricos.

Al eliminar estos caracteres, los valores quedan expresados únicamente con dígitos, permitiendo su correcta conversión a un formato numérico. Este procedimiento es fundamental para garantizar la estandarización de los datos y evitar errores en los cálculos posteriores.

```
try:
```

El uso de la estructura *try* permite manejar de forma controlada situaciones en las que un valor puede generar errores durante su procesamiento. En el contexto de la limpieza de datos, esta estructura se utiliza para intentar convertir valores a formato numérico, aun cuando exista la posibilidad de que algunos datos presenten inconsistencias o formatos inválidos.

La inclusión de este bloque es especialmente relevante cuando se trabaja con datos reales, ya que no todos los registros cumplen con los criterios necesarios para ser transformados correctamente. Al utilizar *try*, se garantiza que el proceso de limpieza continúe ejecutándose incluso cuando se presentan valores problemáticos, evitando la interrupción del flujo del análisis.

```
return float(x)
```

La instrucción *return float(x)* permite devolver el valor procesado en formato numérico decimal. Una vez que el dato ha sido previamente limpiado y se ha eliminado cualquier carácter no numérico, esta conversión garantiza que la información sea interpretada correctamente como un valor cuantitativo continuo.

El uso del tipo de dato *float* es fundamental en el análisis de datos y en los modelos de aprendizaje automático, ya que posibilita la realización de cálculos matemáticos, evaluaciones estadísticas y procesos de optimización. Al asegurar que los valores se encuentren en un formato numérico adecuado, se facilita el aprendizaje del modelo y se reduce el riesgo de errores derivados de tipos de datos incompatibles.

```
except ValueError:
```

El uso de *ValueError* permite gestionar los casos en los que un dato no puede ser convertido correctamente a un valor numérico, incluso después de haber aplicado los procesos de limpieza correspondientes. Este tipo de error suele presentarse cuando el valor contiene caracteres no válidos o formatos inconsistentes que impiden su interpretación como número.

Al contemplar esta situación dentro del proceso de limpieza de datos, se evita que el flujo del análisis se interrumpa por valores atípicos o erróneos. En lugar de detener la ejecución, estos datos son identificados y tratados como valores inválidos, permitiendo continuar con el preprocesamiento de manera ordenada.

Aplicación de limpieza a columnas numéricas problemáticas

En bases de datos provenientes de registros empresariales es común encontrar variables numéricas almacenadas con formatos inconsistentes, por ejemplo: separadores de miles (comas), valores vacíos, texto mezclado con números o símbolos no válidos. Estas irregularidades impiden que las variables sean interpretadas correctamente como numéricas y, por tanto, pueden generar errores en los cálculos estadísticos y en el entrenamiento de modelos de aprendizaje automático.

```
numeric_cols = [ 'Costo_Unitario', 'costo compra', 'costo de venta']
```

Se definió un conjunto de columnas que deben ser tratadas como variables numéricas, ya que representan valores de costos asociados a los repuestos. Estas columnas incluyen el costo unitario, el costo de compra y el costo de venta. Al agruparlas en una sola lista, se facilita la aplicación uniforme del proceso de limpieza y conversión numérica, asegurando que todas las variables monetarias sean estandarizadas antes de ser utilizadas en el análisis y en el modelo predictivo.

```
for col in numeric_cols:
```

Se utiliza para iterar sobre cada columna incluida en la lista *numeric_cols.* Esto permite aplicar el mismo procedimiento de limpieza a todas las variables que representan valores numéricos, como costos unitarios, costos de compra y costos de venta.

En lugar de limpiar cada columna de manera individual, este enfoque automatiza el proceso y garantiza consistencia en el tratamiento de los datos. Cada columna es tomada una por una y procesada de forma idéntica, lo que reduce errores, mejora la eficiencia del preprocesamiento y asegura que todas las variables monetarias queden correctamente estandarizadas antes de ser utilizadas en el análisis y en el modelo de aprendizaje automático.

```
data[col] = data[col].apply(clean_numeric)
```

Esta instrucción aplica la función clean_numeric a todos los valores de la columna seleccionada. Cada dato de la columna es evaluado individualmente para limpiar

caracteres no numéricos, convertir el valor a formato numérico y marcar como nulos aquellos registros que no puedan ser convertidos correctamente.

Al reasignar el resultado a la misma columna, se asegura que la información quede actualizada y estandarizada. Este procedimiento garantiza que las variables de costos contengan únicamente valores numéricos válidos, lo cual es indispensable para realizar cálculos precisos y para el correcto funcionamiento de los modelos de análisis y predicción.

```
int_cols = ['Compras', 'Ventas', 'Stock', 'Stock_Calculado', 'Stock correcto', 'Diferencia']
for col in int_cols:
```

Paso 7: Identificar columnas que representan cantidades enteras.

Corresponden a conteos o unidades físicas y que no deberían contener decimales. Estas variables incluyen compras, ventas, niveles de stock y diferencias de inventario, las cuales son fundamentales para el análisis del comportamiento de los repuestos.

Al agrupar estas columnas en una lista (int_cols), se establece de forma explícita cuáles variables deben manejarse como datos enteros. Esto facilita su tratamiento posterior y permite aplicar de manera uniforme un proceso de validación y conversión de tipo de dato.

```
            data[col] = pd.to_numeric(data[col], errors='coerce').fillna(0).astype(int)
```

Este procedimiento se utiliza para asegurar que las columnas definidas como enteras contengan únicamente valores numéricos válidos y estén correctamente tipificadas como datos enteros. En primer lugar, los valores de cada columna son convertidos a formato numérico, permitiendo identificar y manejar posibles inconsistencias o valores no válidos.

Aquellos registros que no pueden ser interpretados como números son tratados como valores nulos y posteriormente reemplazados por cero. Este criterio se aplica debido a que las variables involucradas representan cantidades discretas, como unidades de compra, ventas o niveles de stock, donde la ausencia de valor puede interpretarse como cero unidades registradas.

Finalmente, los datos son convertidos a tipo entero, garantizando que las variables reflejen correctamente su naturaleza cuantitativa discreta. Este proceso contribuye a mantener la coherencia del conjunto de datos, evita errores en cálculos posteriores y asegura que los modelos de análisis y predicción trabajen con información estructurada y consistente.

```
#Validar o recalcular Stock_Calculado (opcional: usar el dato
existente o recalcular)
# Aquí optamos por confiar en el cálculo lógico: Compras - Ventas
data['Stock_Calculado'] = data['Compras'] - data['Ventas']
```

Paso 8: validación de la variable Stock_Calculado

En esta etapa se realizó una validación de la variable Stock_Calculado, con el fin de asegurar su coherencia lógica dentro del conjunto de datos. Dado que el stock representa la diferencia entre las unidades adquiridas y las unidades vendidas, se optó por recalcular esta variable utilizando la relación directa entre Compras y Ventas.

Este criterio permite corregir posibles inconsistencias presentes en el registro original del inventario y garantiza que el valor del stock refleje fielmente el comportamiento real de los movimientos de entrada y salida de los repuestos. Al basarse en una relación contable lógica, se reduce el riesgo de errores derivados de registros manuales o desactualizados.

La recalculación del stock contribuye a mejorar la calidad de los datos y asegura que esta variable, clave para el análisis y la predicción, sea consistente y confiable antes de ser utilizada en los modelos de aprendizaje automático.

Paso 9: Crear variables derivadas útiles para criticidad

```
data['Impacto_Monetario'] = data['costo de venta']
# o abs(df['costo de venta'])
data['Rotacion'] = data['Ventas']
data['Riesgo_Stock_Negativo'] = (data['Stock_Calculado'] <
0).astype(int)
data['Inconsistencia_Stock'] = np.abs(data['Stock'] -
data['Stock_Calculado'])
```

Se definió la variable Impacto_Monetario, la cual representa el valor económico asociado al repuesto. Esta variable permite reflejar el impacto financiero que tiene cada producto dentro del inventario, siendo un factor clave en la evaluación de criticidad, ya que los repuestos de mayor valor económico implican un mayor riesgo financiero.

La variable Rotación se construyó a partir del número de ventas registradas. Esta variable permite medir el movimiento o frecuencia de salida de los repuestos, indicando qué tan dinámico es un producto dentro del inventario. Una mayor rotación suele asociarse a una mayor importancia operativa.

Adicionalmente, se creó la variable Riesgo_Stock_Negativo, la cual identifica situaciones en las que el stock calculado es menor a cero. Esta variable se expresa de forma binaria, donde el valor uno indica la presencia de riesgo y el valor cero la ausencia del mismo. Su inclusión permite detectar posibles quiebres de inventario o errores en la gestión de existencias, aspectos críticos para la operación de la empresa.

Finalmente, se incorporó la variable Inconsistencia_Stock, que mide la diferencia absoluta entre el stock registrado y el stock recalculado. Esta variable permite identificar desviaciones o inconsistencias en el control del inventario, aportando información relevante sobre la confiabilidad de los registros y su impacto en la toma de decisiones.

Paso 10: Eliminar filas completamente vacías

```
data.dropna(how='all', inplace=True)
```

La función dropna(how='all') se utiliza para eliminar registros que no contienen ningún valor válido dentro del conjunto de datos. Es decir, identifica aquellas filas en las que todas las variables se encuentran vacías o con valores nulos y las descarta del análisis.

El parámetro how='all' indica que la fila será eliminada únicamente cuando todos sus valores sean nulos. De esta manera, se conservan los registros que contienen al menos un dato válido, evitando la pérdida innecesaria de información relevante.

Este procedimiento permite depurar el conjunto de datos eliminando observaciones sin contenido informativo, lo cual mejora la calidad del dataset y garantiza que los modelos de aprendizaje automático se entrenen únicamente con datos reales y útiles.

Proceso 2: K-Means para el nivel crítico

El algoritmo k-means se emplea para identificar de manera objetiva el nivel de criticidad de los repuestos automotrices, la cual surge como la etapa inicial para la ejecución del modelo y su nivel de criticidad, este permite agrupar los repuestos en función a la identificación de patrones reales según sus datos históricos, tales como subgrupo, marca y proveedor.

Aplicación de k-means (clasificación de críticos y no críticos)

Paso 1: Cargar librerías y configurar su visualización

Para iniciar se cargan las librerías necesarias, empezando con *pandas* para la gestión y manipulación de datos estructurados.

```
import pandas as pd
```

Se importa *NumPy* para el manejo eficiente de operaciones numéricas y matemáticas.

```
import numpy as np
```

Para la visualización de los clusters que genera el k-means se aplica.

```
import matplotlib.pyplot as plt
```

Luego se importa el algoritmo a utilizar, en este caso el K-means:

```
from sklearn.cluster import KMeans
```

sklearn.cluster: es un módulo de la librería scikit-learn para generar el algoritmo de agrupamiento.

KMeans: algoritmo del aprendizaje no supervisado para agrupar datos en k grupos.

En el algoritmo K-means es importante estandarizar variables numéricas puesto que al trabajar con k grupos se generan distancias entre los datos y los resultados podrían ser sesgados.

```
from sklearn.preprocessing import StandardScaler
```

sklearn.preprocessing: módulo de la librería scikit-learn.

StandardScaler: se utilizó para la estandarización de las variables numéricas

Su expresión matemática es la siguiente:

$$X_{estandarizado} = \frac{X - \mu}{\sigma}$$

$X_{estandarizado}$: nuevo valor transformado

X: valor original

μ: media de la variable (promedio)

σ: desviación estándar de la variable

El objetivo de la estandarización es que la media sea equivalente a 0 y la desviación estándar 1 para comparar variables con escalas diferentes.

Se utiliza ACP (Análisis de Componentes Principales) para disminuir y visualizar la dimensionalidad de los datos, no para clasificar.

```
from sklearn.decomposition import PCA
```

sklearn.decomposition: este módulo tiene técnicas de descomposición y minimización de las dimensiones.

Por consiguiente, se importan dos métricas para evaluar la calidad del cluster:

```
from sklearn.metrics import silhouette_score,
                            calinski_harabasz_score
```

silhouette_score: evalúa que tan bien están asignados los repuestos en cada cluster.

$$s = \frac{b_i - a_i}{max(a_i, b_i)}$$

a: distancia promedio de *i* hacia todos los demás puntos de su cluster.

b: distancia promedio de *i* hacia todos los puntos del grupo vecino más cercano.

calinski_harabasz_score: evalúa la separación entre los clusters y la compactación que hay dentro de cada cluster.

Paso 2: Cargar el conjunto de datos

Una vez cargadas las librerías se procede a cargar los datos con los cuales se va a trabajar.

```
from google.colab import files

uploaded = files.upload()
```

Leemos los datos una vez que ya están limpios

```
#Leemos los datos limpios

df = pd.read_excel("Repuestos_automotores_dataclean",

sheet_name="Sheet1")

df
```

Paso 3: Selección de variables para el clustering

Se definen las variables que se van a utilizar para hacer el clustering y así definir la criticidad.

```
features = ['Rotacion', 'Impacto_Monetario',

'Stock_Calculado', 'Inconsistencia_Stock']

X = df[features].copy()
```

features: son las características o variables independientes que se van a utilizar para hacer los clusters mediante el k-means. Estas variables son aquellas que influyen para determinar la criticidad de un repuesto.

Rotacion: es la frecuencia con la que un repuesto es demandado. Mayor rotación, mayor criticidad.

Impacto_Monatario: valor económico asociado al repuesto.

Stock_Calculado: es la cantidad de repuestos disponibles en el inventario. Poco st ock, aumenta la criticidad.

Inconsistencia_Stock: es la diferencia entre el el stock registrado y el stock esperado.

Se crea el conjunto de datos o DataFrame para el cluster, denominado como X, el cual contiene únicamente las variables seleccionadas anteriormente.

```
X = df[features].copy()
```

.copy(): esto previene las modificaciones accidentales y mantiene el DataFrame original.

Paso 4: Estandarización de variables

En el algoritmo K-means al hacer agrupaciones de características, es un modelo que se basa en distancias, mientras más grande sean los valores de la variable, tendrá más influencia en la formación de los clusters. Es por esto que, se realiza la estandarización de los valores para que el modelo aprenda de manera correcta, evitando el sesgo y una clasificación poco representativa.

Se empieza por aplicar la estandarización para que todas las variables influyan por igual, mediante:

```
scaler = StandardScaler()
```

Para el ajuste y transformación de los datos:

```
X_scaled = scaler.fit_transform(X)
```

fit (X): aprende la distribución de la información y calcula la media y la desviación estándar de cada variable.

transform(X): transforma las variables originales a la estandarización.

Paso 5: Aplicación del algoritmo K-means

Una vez realizada la estandarización de las variables y sus valores, se clasifican los repuestos según su nivel de criticidad. Por lo que se crea un código con sus respectivas especificaciones:

```
kmeans = KMeans(n_clusters=3, random_state=42, n_init=10)
```

n_clusters=3: divide el algoritmo en 3 clusters distintos basados en su similitud.

random_state=42: es la semilla de aleatoriedad del modelo, asegura que los datos se reproduzcan correctamente al ejecutar.

n_init=10: el modelo se ejecuta 10 veces con otros centroides para que se pueda seleccionar el que tenga una menor suma de distancias.

Se ajusta el modelo para darle una asignación a cada cluster.

```
df['Cluster_KMeans'] = kmeans.fit_predict(X_scaled)
```

df['Cluster_KMeans']: aquí se guardan los resultados.

predict(X_scaled): se le asigna a cada repuesto un cluster (0, 1, 2).

fit(X_scaled): calcula los centroides y minimiza las distancias del repuesto y el centroide.

Paso 6: Obtención e interpretación de centroides

Para interpretar los resultados del agrupamiento, se calcula el centroide promedio de cada cluster y vuelve nuevamente en escala original para su interpretación. Este código hace la extracción de los centroides finales de las agrupaciones que generó el k-means.

```
centroids_scaled = kmeans.cluster_centers_
```

Se aplica la transformación inversa de *StandardScaler*, y de este modo recuperar e interpretar los valores originales de las variables para terminar la criticidad de cada cluster.

```
centroids_original = scaler.inverse_transform(centroids_scaled)
```

En un DataFrame estructurado se organizan los centroides. Las filas son los clusters y las columnas una variable relevante.

```
centroid_df = pd.DataFrame(centroids_original, columns=features)
```

Paso 7: Asignación de niveles de criticidad

Asignar etiquetas: *"Alto"*, *"Medio"*, *"Bajo"* según la rotación y el impacto

Se empieza por ordenar clusters según la rotación descendente.

```
centroid_df['cluster_id'] = centroid_df.index
```

'cluster_id': almacena el identificador original de los clusters, ya sea 0, 1 o 2.

Se ordenan los clusters según su rotación de forma descendente. Los que constan con mayor rotación son los que generalmente son los más críticos.

```
centroid_df_sorted = centroid_df.sort_values(by='Rotacion',
ascending=False).reset_index(drop=True)
```

.reset_index(drop=True): se reinicia el índice para obtener un mejor orden.

drop=True: es un parámetro que evita el riesgo de que el índice se transforma en una columna innecesaria.

Mapeo de clusters según su nivel de criticidad: el primero = Alto, segundo = Medio, tercero = Bajo

```
cluster_stats = df.groupby('Cluster_KMeans')[['Rotacion',
'Impacto_Monetario']].mean()
```

cluster_stats: indica el cluster al que se le asignó a cada repuesto.

.groupby('Cluster_KMeans'): se agrupa el DataFrane por cluster.

[['Rotacion', 'Impacto_Monetario']]: variables seleccionadas para determinar la criticidad.

.mean(): dentro de cada cluster se saca el promedio de cada variable.

Se ordenan de mayor a menos los clusters por el promedio de su rotación

```
critical_order = cluster_stats.sort_values(by='Rotacion',
                                ascending=False).index
```

critical_order: índice ordenado con los clusters según su nivel de criticidad.

.sort_values(by='Rotacion', ascending=False): para la jerarquización.

Rotacion: rotación promedio

ascending=False: orden de mayor a menor

.index: guarda los números del cluster según su criticidad.

Se hace la definición del mapeo de etiquetas, mediante la creación de un diccionario que asigna una etiqueta de criticidad a los 3 clusters. Se basa en el orden generado gracias a los centroides y al promedio de la rotación por agrupamiento.

```
label_map = {centroid_df_sorted.loc[0, 'cluster_id']: 'Bajo',
             centroid_df_sorted.loc[1, 'cluster_id']: 'Medio',
             centroid_df_sorted.loc[2, 'cluster_id']: 'Alto'}
```

label_map: librería de Python.

centroid_df_sorted: es la tabla de centroides ordenada por rotación.

cluster_id: cluster real

.loc: extrae un valor de una tabla según la fila-columna.

Seguidamente se crea una columna llamada Criticidad.

```
df['Criticidad'] = df['Cluster_KMeans'].map(label_map)
```

df['Cluster_KMeans']: la columna que contiene los resultados del modelo.

.map: función que toma los valores y los reemplaza con la librería a aplicar.

label_map: librería para la traducción de los números a palabras.

Paso 8: Validación del modelo

Se aplican métricas de validación del modelo.

```
sil_score = silhouette_score(X_scaled, df['Cluster_KMeans'])
```

X_scaled: datos estandarizados

df['Cluster_KMeans': asignación de clusters para cada repuesto.

silhouette_score: métrica matemática que evalúa la calidad de los clusters formados, puesto que, al ser un algoritmo de aprendizaje No supervisado, no tiene una variable real para comparase.

Por último, se utiliza una métrica que evalúa la parte interna del algoritmo, midiendo si la separación y la compactación entre los grupos están bien entre sí.

```
ch_score = calinski_harabasz_score(X_scaled,
                                   df['Cluster_KMeans'])
```

ch_score: es el valor obtenido del Calinski–Harabasz que indica que tan bien fueron generados los grupos.

X_scaled: datos estandarizados con media 0 y desviación estándar 1.

df['Cluster_KMeans']: vector que indica de que grupo es cada repuesto.

calinski_harabasz_score: compara la dispersión entre clusters (B) y la dispersión dentro de los clusters (W).

$$CH = \frac{B/(k-1)}{W/(N-k)}$$

k: clusters

n: número total de observaciones

Reordenar columnas

```
cols = list(df.columns)
```

df.columns: está compuesto por ['Repuesto', 'Rotacion', 'Impacto_Monetario', 'Stock_Calculado', 'Inconsistencia_Stock', 'Cluster_KMeans', 'Criticidad'].

```
cols.insert(0, cols.pop(cols.index('Criticidad')))
```

cols.index('Criticidad'): sirve para buscar la posición de la variable *Criticidad* dentro de la lis *cols.*

cols.pop(): extrae y elimina la columna criticidad se su lugar original.

cols.insert(0,..): inserta la columna Criticidad en la primera columna, es decir en la posición 0.

```
cols.insert(1, cols.pop(cols.index('Cluster_KMeans')))
```

cols.index('Cluster_KMeans'): halla en donde está la columna.

cols.pop(…): elimina a *'Cluster_KMeans'* de su ubicación inicial y la devuelve.

cols.insert(1, ...): inserta *'Cluster_KMeans'* en la segunda columna, es decir la posición 1.

Paso 9: Análisis descriptivo de resultados

Se crea un DataFrame con el nuevo orden de las columnas, esta reorganización es únicamente de manera visual.

```
df = df[cols]
```

En resumen, del modelo.

```
summary = df['Criticidad'].value_counts().reindex(['Bajo',
          'Medio', 'Alto'], fill_value=0)
```

df['Criticidad']: contiene la clasificación de los repuestos. Es la variable que se quería hallar.

.value_counts(): cuentas las veces que se repite una categoría.

.reindex(['Bajo', 'Medio', 'Alto']: re ordena los índices.

fill_value=0: si hay inexistencias en alguna categoría, se completa con 0.

Se imprime un mensaje que informa.

```
print(f"Clustering completado con k=3.")
```

k=3: indica *Bajo, Medio, Alto*

Se escribe el mensaje que será el encabezado del resumen.

```
print(f"Distribución de criticidad:")
```

Luego recorre el objeto **summary** (resumen) según clave-valor.

```
for level, count in summary.items():
```

Se imprime cada categoría de manera ordenada.

```
print(f"{level}: {count} ítems")
```

Por lo tanto, el resumen queda de la siguiente manera:

```
Clustering completado con k=3.
Distribución de criticidad:
```

- Bajo: 6 ítems
- Medio: 240 ítems
- Alto: 4109 ítems

Los resultados de clustering evidencian que existe un total aproximado de 4355 repuestos, y de estos, existe un predominio del 94% de repuestos altamente críticos, por lo que la mayoría de las autopartes generar una alta rotación del inventario.

Paso 10: Reducción de dimensionalidad mediante ACP

Se hace una proyección en 2D con fines visuales para reducir la dimensionalidad.

```
pca = PCA(n_components=2)
```

n_components=2: reduce los datos solamente en dos dimensiones.

```
X_pca = pca.fit_transform(X_scaled)
```

X_pca: matriz de datos.

fit(): reduce los datos.

transform(): hace la transformación.

Paso 11: Visualización gráfica de los resultados

Gráfico 1: Proyección PCA

Para crear la figura

```
plt.figure(figsize=(8, 6))
```

8, 6: pulgadas de ancho y 6 pulgadas de alto.

Definir por colores para una mejor inyerepretación:

```
colors = {'Bajo': 'green', 'Medio': 'orange', 'Alto': 'red'}
```

Iterar según los niveles de criticidad:

```
for level in ['Bajo', 'Medio', 'Alto']:
mask = df['Criticidad'] == level
```

mask: vector booleano de True y False. Actúa como un filtro.

`plt.scatter(:` dibuja los puntos

`X_pca[mask, 0],:` selecciona el componente principal 1 (PC1)

`X_pca[mask, 1],:` selecciona el componente principal 2 (PC2)

`c=colors[level],` # <-- ¡Asignación explícita de color!

`label=level,:` sirve para identificar los grupos

`alpha=0.6,:` evitar la transparencia de los puntos

`s=20):` es el tamaño de los puntos

Para el título del gráfico

```
plt.title('K-means Clusters (Proyección PCA)')
plt.xlabel(f'PC1({pca.explained_variance_ratio_[0]:.1%}
varianza)')
plt.ylabel(f'PC2({pca.explained_variance_ratio_[1]:.1%}
varianza)')
```

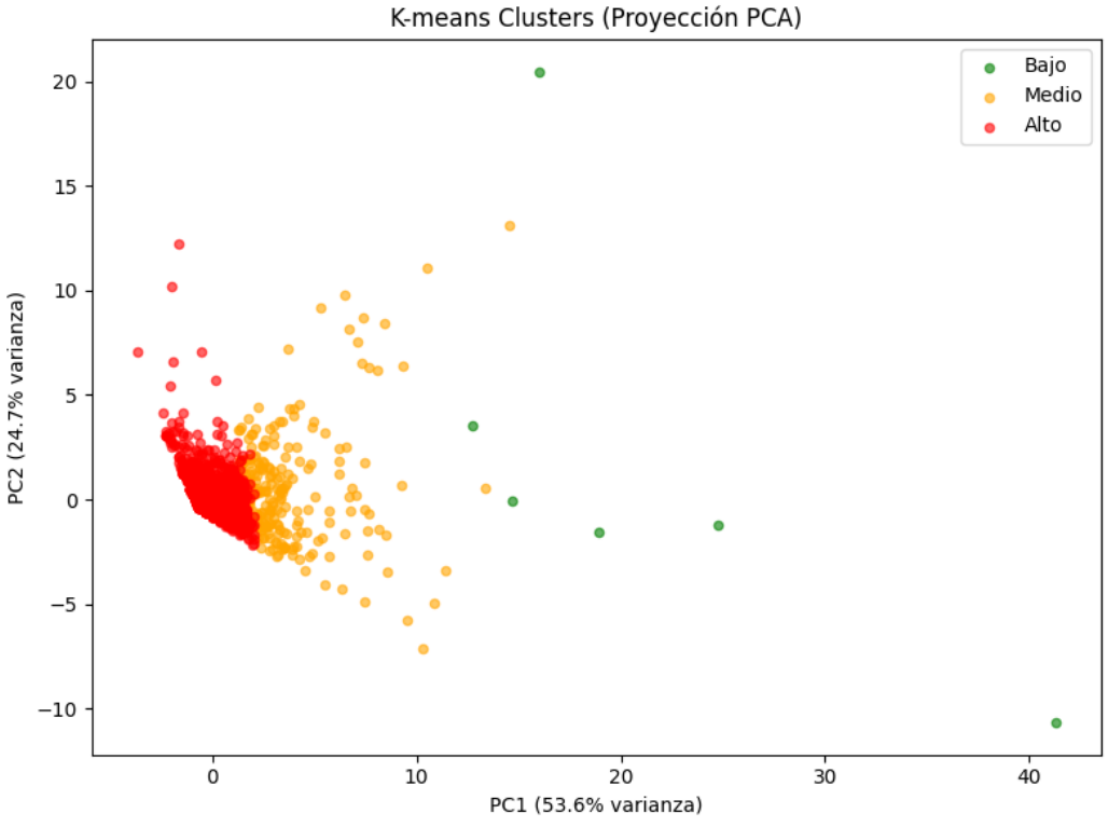
`plt.legend():` muestra el nivel ya sea Bajo, Medio, Alto

`plt.tight_layout()`: ajusta y mejora el diseño

`plt.savefig("kmeans_pca.png", dpi=150)`: guardar el gráfico con buena calidad.

`plt.show()`: visualizar el grafico en pantalla

Figura 6 Proyección de Clusters K-Means mediante el Análisis de Componentes Principales



La proyección PCA evidencia una estructura de agrupamiento, donde los repuestos de alta criticidad presentan una elevada cohesión, por otro lado, los de baja criticidad aparecen claramente separados del resto. El grupo de criticidad media actúa como una zona de transición, lo que refleja la naturaleza gradual del comportamiento de los repuestos.

Gráfico 2: Perfiles normalizados de clusters

Se agrupan los repuestos según su nivel de criticidad.

```
profile = df.groupby('Criticidad')[features].mean()
```

Se normalizan las variables entre 0 y 1.

```
profile_norm = (profile - profile.min()) / (profile.max() - profile.min())
```

```
fig, ax = plt.subplots(figsize=(8, 5))
```

`fig`: crea la figura

`ax`: crea un eje para dibujar el grafico.

Se definen las posiciones del eje X:

```
x = np.arange(len(features))
```

`width = 0.25:` ancho de las barras (3)

Se dibujan las barras para el cluster Bajo:

```
ax.bar(x - width, profile_norm.loc['Bajo'], width, label='Bajo',
                    color='green', alpha=0.7)
```

`x - width:` desplazar a la izquiera

Se dibujan las barras para el cluster Medio:

```
ax.bar(x, profile_norm.loc['Medio'], width, label='Medio', color='orange', alpha=0.7)
```

Se dibujan las barras para el cluster Alto:

```
ax.bar(x, profile_ ax.bar(x + width, profile_norm.loc['Alto'], width, label='Alto', color='red', alpha=0.7)
```

Se configuran las etiquetas del eje X

```
ax.set_xticks(x)
```

```
ax.set_xticklabels(features, rotation=45, ha='right')
```

`ax.set_ylabel('Valor normalizado (0-1)')`: son valores comparativos

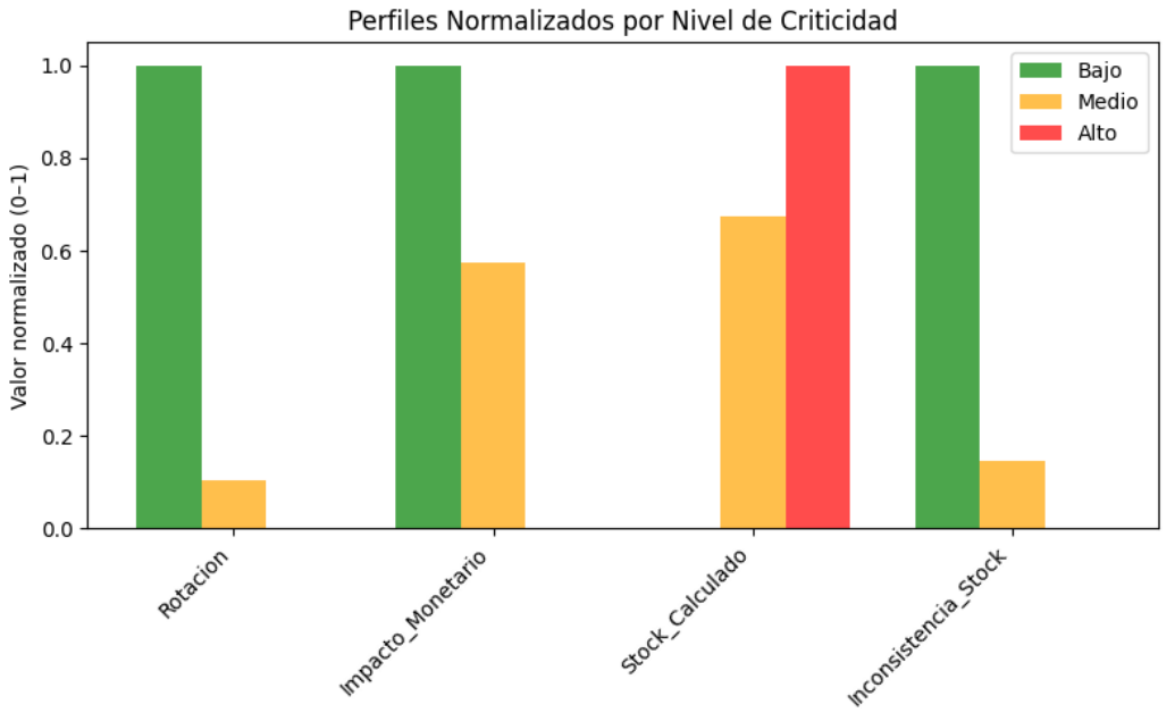
`ax.set_title('Perfiles Normalizados por Nivel de Criticidad')`: título del gráfico

`ax.legend()`: identificar los clusters

```
plt.tight_layout()
plt.savefig("cluster_profiles.png", dpi=150)
```

`plt.show()`: visualizar el gráfico

Figura 7 Perfiles normalizados de Clusters



En el análisis de perfiles normalizados se evidencian diferencias claras entre los 3 niveles de criticidad. Los ítems altamente críticos presentan baja rotación y bajo impacto monetario, pero requieren mayores niveles de stock, lo que refleja su importancia operativa. En contraste, los ítems de baja criticidad muestran alta rotación e impacto monetario, pero menor necesidad de stock de seguridad, confirmando la coherencia del proceso de clustering aplicado.

Gráfico 3: Conteo de ítems por cluster

Empieza con el conteo de los ítems según su nivel crítico.

```
counts = df['Criticidad'].value_counts().reindex(['Bajo', 'Medio', 'Alto'])
```

`value_counts().:` cuenta los ítems de cada categoría.

`reindex(['Bajo', 'Medio', 'Alto']):` genera un orden logico

```
plt.figure(figsize=(6, 4))
```

6: ancho

4: alto

Se crea el código de barras mediante:

```
bars = plt.bar(counts.index, counts.values, color=['red', 'orange', 'green'], alpha=0.7)
```

`counts.index:` son las etiquetas del eje X

`plt.bar():` crea el gráfico de barras

`Alpha=0.70:` mejor estética y control de la transparencia

`for bar in bars:` objeto que contiene a todas las barras

Posición del eje X

`plt.text(bar.get_x() + bar.get_width()/2:` coloca el texto centrado sobre la barra

Posición del eje Y:

`, bar.get_height() + 20,:` coloca el texto un poco arriba sobre la barra

Para el texto:

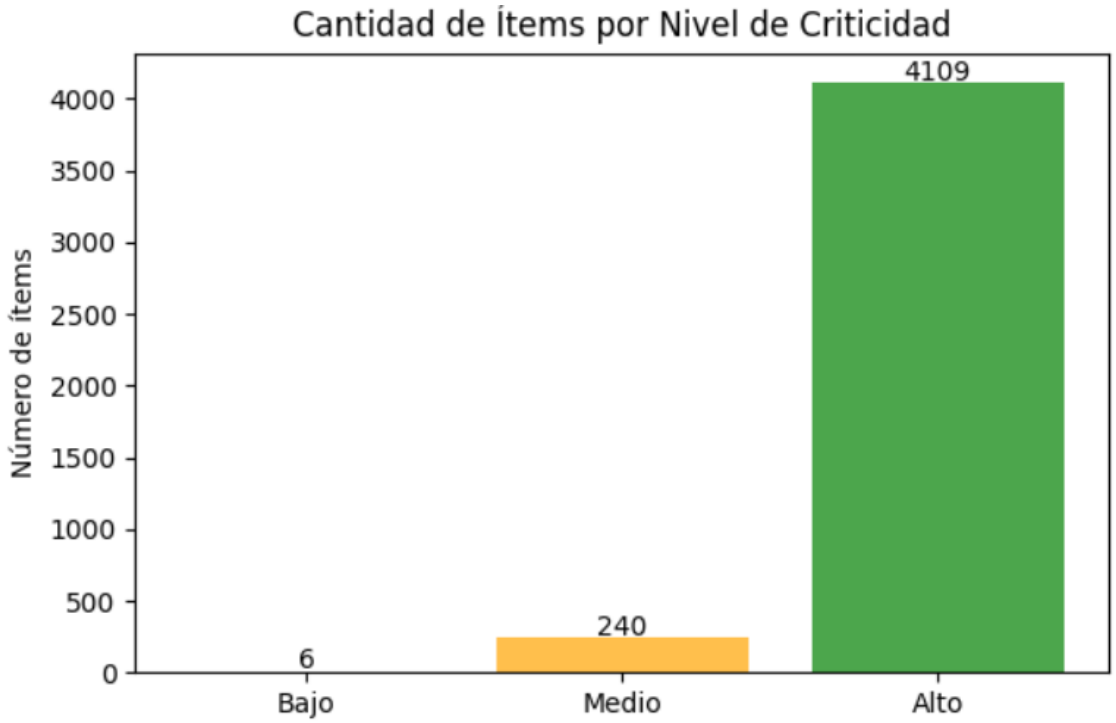
`str(int(bar.get_height())),` muestra el número total de ítems

`ha='center'):` centra el texto de forma horizontal

```
plt.title('Cantidad de Ítems por Nivel de Criticidad')
plt.ylabel('Número de ítems')
```

```
plt.tight_layout()
plt.savefig("cluster_counts.png", dpi=150)
plt.show()
```

Figura 8 Conteo de ítems por clusters



El gráfico de la distribución por criticidad según la cantidad de ítems evidencia que la mayoría de los repuestos analizados se clasifican como altamente críticos, mientras que un grupo reducido presenta baja criticidad. Esta distribución refleja la naturaleza del inventario, es decir que, es caracterizado por repuestos de baja rotación, pero alto impacto operativo, lo cual valida la coherencia del modelo K-Means aplicado.

Gráfico 4: Boxplots por característica

`fig, axes = plt.subplots(2, 2, figsize=(12, 8))`

`fig:` figura principal

`axes:` matriz de 2 filas x 2 columnas

`axes = axes.ravel():` convierte y aplaza los ejes de la matriz

`for i, col in enumerate(features):` devuelve un índice con el nombre de su variable.

Para cada nivel se filtra el DataFrame y solamente extrae la variable actual.

```
data = [df[df['Criticidad'] == lvl][col].values for lvl in ['Alto', 'Medio', 'Bajo']]
```

Para crear el boxplot:

```
bp = axes[i].boxplot(data, tick_labels=['Bajo', 'Medio', 'Alto'], patch_artist=True)
```

`tick_labels`: son nombres del eje X

`patch_artist`=True: permite colorear las cajas

Establecer los colores de cada casilla manualmente

```
for patch, color_name in zip(bp['boxes'], ['Bajo', 'Medio', 'Alto']): patch.set_facecolor(colors[color_name])
```

`bp['boxes']`: lista de cajas del boxplot

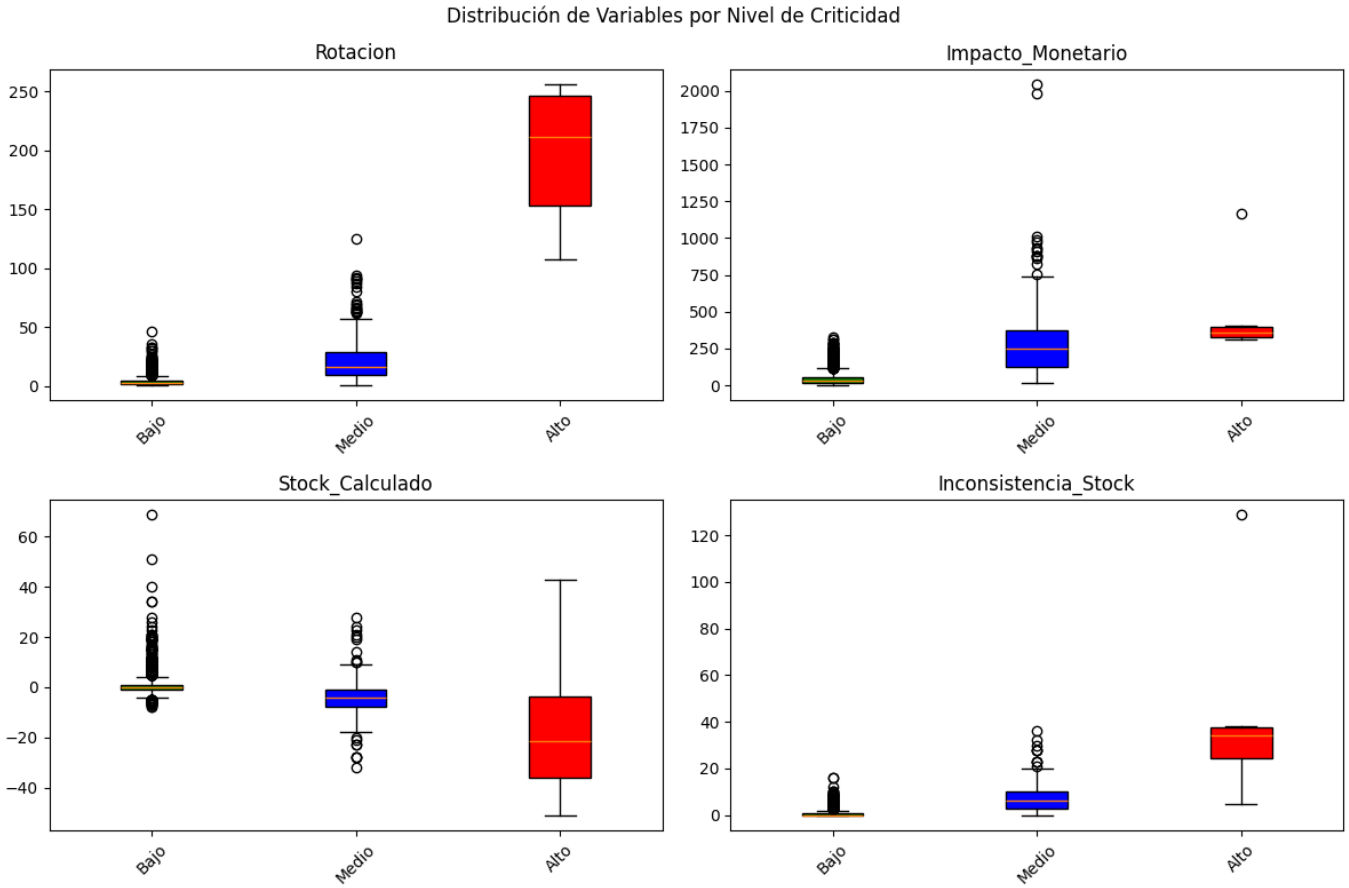
`zip(...)`: asocia cada caja con su nivel

`set_facecolor()`: asigna el color correspondiente

Para el título del gráfico

```
axes[i].set_title(f'{col}')
axes[i].tick_params(axis='x', rotation=45)
plt.suptitle('Distribución de Variables por Nivel de Criticidad')
plt.tight_layout()
plt.savefig("boxplots_features.png", dpi=150)
plt.show(
```

Figura 9 Boxplots por característica



Los diagramas de caja evidencian diferencias significativas en la distribución de las variables analizadas según el nivel de criticidad. Se observa una separación clara entre los clusters en términos de rotación, impacto monetario, stock calculado e inconsistencia de stock, lo que confirma la efectividad del algoritmo K-Means para clasificar los repuestos de acuerdo con su comportamiento operativo.

Proceso 3: Aplicación del algoritmo Random Forest Regressor

La aplicación del algoritmo Random Forest o también conocido como Bosques Aleatorios, se emplea como un método de ensamble que combina múltiples árboles de decisión, su objetivo es optimizar su capacidad predictiva y generar un modelo con robustez y así reducir el sobreajuste. En este caso, al entrenar múltiples variables interrelacionadas un bosque siempre tiene que tener mejor correlación que el árbol, lo cual permite captar decisiones que anteriormente no captaba y reducir la varianza.

Para la regresión, la predicción se da mediante el promedio de todas las predicciones de cada árbol individualmente, lo cual suaviza el resultado generando estabilidad y reduciendo el error.

Aplicamos Random Forest para la predicción de compras a niveles promedios

Se importa el Random Forest Regressor como el modelo predictivo a utilizar. Este algoritmo es utilizado cuando la variable objetivo es numérica continua, en este caso la cantidad de compras.

Paso 1: Importación de librerías

```
from sklearn.ensemble import RandomForestRegressor
```

La siguiente función permite dividir el conjunto de datos, para que el modelo aprenda: datos de entrenamiento, y para la evaluación del modelo: datos de prueba.

```
from sklearn.model_selection import train_test_split
```

El conjunto de datos fue dividido mediante la función `train_test_split` del modelo `sklearn.model_selection` es un paquete de la librería scikit-learn utilizada con el fin de separar, validar y reducir el riesgo del overfitting.

Las métricas de evaluación del modelo de regresión son el Mean Absolute Error (MAE) y el r^2_score (R^2).

`mean_absolute_error:` mide el promedio de los errores de una predicción para evaluar la efectividad de una regresión, mediante la diferencia absoluta promedio entre el valor real y el valor predicho.

`r2_score:` mide el grado de ajuste del modelo, en base a la proporción de la varianza en la variable dependiente a partir de las variables independientes. Su definición matemática es la siguiente:

$$R^2 = 1 - \frac{\sum_i^N (y_i - \hat{y}_i)^2}{\sum_i^N (y_i - \bar{y})^2}$$

```
from sklearn.preprocessing import LabelEncoder
```

LabelEnconder es una herramienta de scikit-learn utilizada para convertir datos categóricos en valores numéricos, de la siguiente manera:

```
Alto=2, Medio=1, Bajo=0
```

Por consiguiente, se utilizó una librería para gestionar y ocultae las advertencias y que el modelo se ejecute de una manera más ordenada y limpia sin afectar su resultado.

```
import warnings
warnings.filterwarnings("ignore")
```

Para la visualización de los resultados la importa la librería de Malplotlib, la cual permite generar gráficos para analizar datos e interpretar el comportamiento del algortimo.

```
import matplotlib.pyplot as plt
```

Finalmente, la última librería a importar para desarrollar el Bosque Aleatorio es:

```
from sklearn.tree import plot_tree
```

Esta función permite visualizar un árbol de decisión individual ya entrenado, es decir uno que forme parte del Random Forest, con el objetivo de interpretar las decisiones y el comportamiento del modelo.

Paso 2: Cargar el conjunto de datos

Una vez cargadas las librerías se procede a cargar los datos a Python para comenzar con la ejecución del modelo. Para ello se utiliza el siguiente código:

```
from google.colab import files

uploaded = files.upload()
```

A continuación, se realiza una lectura de los datos:

```
df =
pd.read_excel("Repuestos_automotores_con_Criticidad.xlsx",
sheet_name="Sheet1")
```

pd.read_excel: este código permite trabajar con archivos del Excel, para observar las filas y columnas y conserva los datos como una tabla estructurada.

"Repuestos_automotores_con_Criticidad.xlsx": nombre del archivo que contiene la matriz de datos.

sheet_name = "Sheet1": se especifica que la hoja 1 es la que contiene los datos que se quieren visualizar.

df: se carga el DataFrame para visualizar la tabla.

Figura 10 DataFrame

	Criticidad	Cluster_KMeans	Codigo	SubGrupo	Proveedor	Marca	Descripcion	Costo_Unitario	Compras	Ventas	...	costo compra	costo de venta	categorias	Stock_Calculado	Stock correcto	Diferencia	Impacto_Monetario	Rotacion	Riesgo_Stock_Negativo	Inconsistencia_Stock
0	Bajo	2	2349	REFRIGERANTE	PROMESA	PRESTONE	PRESTONE.	4.5450	299	256	...	1358.96	1163.52	Lubricantes Filtros y refrigerantes	43	0	5	1163.52	256	0	5
1	Bajo	2	3051	BUJIAS	L.HENRIQUEZ & CIA S.A	NGK	CHEV AVEO NISSAN B13 SWIFT STEEM SZ	1.5450	210	248	...	324.45	383.16	Sistema Eléctrico	-38	0	38	383.16	248	1	38
2	Bajo	2	3050	BUJIAS	L.HENRIQUEZ & CIA S.A	NGK	CHEV CORSA LUV DMAX CHEVY	1.3000	190	241	...	247.00	313.30	Sistema Eléctrico	-51	0	129	313.30	241	1	129
3	Bajo	2	3049	BUJIAS	L.HENRIQUEZ & CIA S.A	NGK	CHEV SAIL 1.4 N200 N300 BEAT SPARK GT	1.8000	150	181	...	270.00	325.80	Sistema Eléctrico	-31	0	31	325.80	181	1	31
4	Bajo	2	2949	SILICON	PROMESA	PERMATEX	GRIS PERMATEX	2.3129	143	144	...	330.74	333.06	Siliconas y Limpiadores	-1	0	22	333.06	144	1	22
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
4350	Alto	0	3320	CILINDRO EMBRAGUE	IMPORTADORA MAGIATY CIA. LTDA.	MGT	AUX CHEV LUV 2.5	6.6597	1	1	...	6.66	6.66	Embrague	0	1	0	6.66	1	0	0
4351	Alto	0	2867	BOBINA ENCENDIDO	IMPORTADORA MAGIATY CIA. LTDA.	DRAPA	CHEV, ASTRA, ZAFIRA	10.5544	1	1	...	10.55	10.55	Sistema Eléctrico	0	1	0	10.55	1	0	0
4352	Alto	0	2868	BOBINA ENCENDIDO	IMPORTADORA MAGIATY CIA. LTDA.	DRAPA	SUZ. S-CROSS, VITARA SZ.	10.5541	4	1	...	42.22	10.55	Sistema Eléctrico	3	1	0	10.55	1	0	0
4353	Alto	0	2871	PUNTA INT	IMPORTADORA MAGIATY CIA. LTDA.	GMSGS	CHEV SAIL 1.4	19.2000	4	1	...	76.80	19.20	Direccion y Suspension	3	1	0	19.20	1	0	0
4354	Alto	0	2309	EMPAQUE TAPA VALVULA	IMPORTADORA CERON	ONNURI	HY EXCEL.	1.1300	1	1	...	1.13	1.13	Repuestos de motor	0	1	0	1.13	1	0	0

4355 rows × 21 columns

Paso 3: Selección de variables predictoras y variable objetivo

Posteriormente visualizando la tabla de datos, se seleccionaron las variables más relevantes para ejecutar el modelo:

```
features =
['Criticidad','Ventas','Stock_Calculado','Impacto_Mone
tario','Inconsistencia_Stock','categorias']
target = 'Compras'
X = df[features].copy()
y = df[target]
```

features: son las características o variables independientes que se van a utilizar para entrenar al modelo. Estas variables son aquellas que influyen en la variable "Compras".

`Criticidad:` nivel de importancia del repuesto

`Ventas:` comportamiento histórico

`Stock_Calculado:` disponibilidad estimada del inventario

`Impacto_Monetario:` efecto económico

`Inconsistencia_Stock:` desviación del inventario

`Categorias:` clasificación de los puestos

target: definimos la variable objetivo o dependiente, es decir aquella que el modelo va a predecir.

"Compras": es la variable numérica continua

X = df[features]: del DataFrame se selecciona únicamente las columnas que fueron definidas como ***features***, las cuales son receptadas como variables de entrada denominada ***X***.

.copy(): esta función genera una copia para evitar alterar el DataFrame original.

y = df[target]: en esta función se selecciona la columna ***Compras*** del DataFrame y se recepta su información en la variable de salida denominada ***Y***.

Paso 4: Codificación de las variables categóricas

Se comenzó con la codificación de la variable ordinal *Criticidad*. Para ello, se crea un diccionario de mapeo que almacena y representa una asignación clave-valor a los datos. En este caso, a cada categoría de Criticidad se le asigna un valor numérico de la siguiente manera:

```
crit_map = {'Bajo': 0, 'Medio': 1, 'Alto': 2}
```

Esto se debe a que la Criticidad tiene una escala natural u ordinal y el modelo tiene que aprender de forma jerárquica que *Alto* tiene más peso que *Medio* y *Bajo*. Por consiguiente, se utiliza el siguiente código que reemplaza los valores de texto de la variable Criticidad por los que son numéricos, los cuales son definidos con `crit_map`.

```
X['Criticidad'] = X['Criticidad'].map(crit_map)
```

`.map:` recorre la columna *Criticidad* y toma el valor *Alto*, *Medio* y *Bajo* para luego buscarlo en el diccionario creado por `crit_map` y lo reemplaza por el valor numérico correspondiente. Por lo tanto, la columna *Criticidad* queda netamente con valores numéricos.

Paso 5: Codificación de variable categórica

Luego se realiza la codificación de la variable categórica *target encoding*, (mejor que *one–hot* para árboles y pocos datos por categoría):

```
cat_means = df.groupby('categorias')['Compras'].mean()
```

cat_means: calcula el promedio de las compras para cada categoría.

df: contiene a las filas y columnas, incluyendo a las categorías y las compras.

.groupby('categorias'): agrupa los datos según el tipo de categoría.

“Compras”: para cada categoría solo se enfoca en cuánto se compra.

.mean(): calcula el promedio de la variable *Compras* dentro de cada categoría.

Las categorías se transforman en un valor numérico, reemplazandolas por su promedio de compras:

```
X['categorias_encoded'] = X['categorias'].map(cat_means)
```

Para la eliminar la columna categórica original:

```
X = X.drop(columns=['categorias'])
```

X: contiene las variables independientes, es decir las numéricas originales, la Criticidad codificada, las categorias en texto y numéricas.

.drop: función de la librería Pandas para eliminar la columnas del DataFrame.

columns = ['categorias']: se especifica *categorias* porque esa se va a eliminar, puesto que ya existe 'categorias_encoded'.

Paso 6: División del conjunto de datos

Para la división del entrenamiento y prueba:

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

X: variables explicativas/independientes

y: variable objetivo/dependiente o valor real

X_train: son las variables explicativas para el entrenamiento.

X_test: son las variables explicativas para la prueba.

y_train: son los valores reales de la variable *Compras* en el entrenamiento.

y_test: son los valores reales de la variable *Compras* para la prueba.

train_test_split (X, y, ...): es una función de sklearn.model_selection que hace la partición aleatoria de los datos.

test_size=0.2: el 20% de los datos con para la prueba, es decir el 80% restante son destinados para el entrenamiento.

random_state=42: este parámetro es aquel que fija la semilla del generador de manera aleatoria.

Paso 6: Configuración del modelo Random Forest

Para entrenar el Random Forest

```
rf = RandomForestRegressor(n_estimators=200, max_depth=5,
min_samples_split=10, min_samples_leaf=5, random_state=42,
                           n_jobs=-1)
```

rf = RandomForestRegressor: se crea un objeto que represente el modelo.

n_estimators=200: es el número de árboles del bosque.

max_depth=5: la profundidad de cada árbol tiene máximo 5 niveles.

min_samples_split=10: se estableció que el nodo se divide únicamente si tiene al menos 10 datos.

min_samples_leaf=5: se estableció que cada hoja del árbol debe tener al menos 5 datos.

random_state=42: se estableció la semilla aleatoria para que se reproduzca el resultado.

n_jobs= -1: es el número de núcleos a utilizar.

Paso 8: Entrenamiento del modelo

Luego el modelo aprende, analizando la relación entre *X_train* y *y_train.*

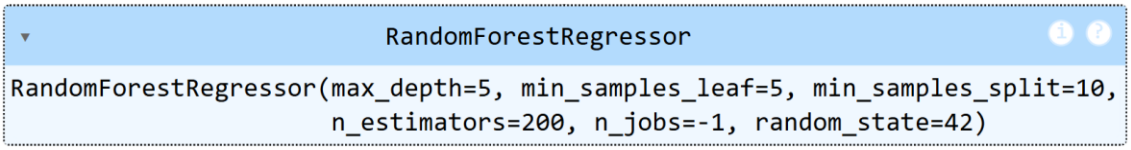
```
rf.fit(X_train, y_train)
```

Capítulo III

Resultados

En este punto el modelo construyó árboles de decisión y aprendió patrones.

Figura 11 Algoritmo Random Forest Regressor



En esta etapa se generan las predicciones y las métricas con el siguiente código.

```
y_pred = rf.predict(X_test)
```

y_pred: contiene los valores predichos de compras.

rf.predict: genera la predicción

X_test: contiene las variables explicativas que no fueron utilizadas durante el modelo.

```
mae = mean_absolute_error(y_test, y_pred)
```

mae = mean_absolute_error: es el promedio de las diferencias absolutas.

y_test: valores reales

y_pred: valores predichos

```
r2 = r2_score(y_test, y_pred)
```

r2 = r2_score: mide la proporción de la variabilidad de *Compras*.

Finalmente se muestra un mensaje informando que el modelo ha sido entrenado:

```
print("Modelo entrenado.")
```

De la misma manera se muestra el valor del MAE:

```
print(f"MAE (Error absoluto medio): {mae:.2f} unidades")
```

{mae:.2f}: se redondea a 2 decimales para facilitar la lectura.

Por última de muestrea el valor obtenido del R^2:

```
print(f"R² (Coeficiente de determinación): {r2:.3f}")
```

{r2:.3f}: se redondea a 3 decimales ya que es una medida estadística.

Una vez cargado los anteriores códigos se obtuvo lo siguiente:

```
Modelo entrenado
MAE (Error absoluto medio): 1.18 unidades
R² (Coeficiente de determinación): 0.943
```

MAE = 1.18 unidades

Según el MAE obtenido, en promedio las predicciones obtenidas por el modelo se desvían 1.18 unidades de compra respecto a los valores reales. Este error es considerado bajo e indica que las predicciones se acercan a los valores reales.

R^2 = 0.943

Según el coeficiente de determinación obtenido, indica que el modelo puede explicar el 94.3% de la variabilidad de la variable *Compras*, al ser un valor cercano al 1 se interpreta como un excelente nivel de ajuste.

Importancia de las variables

En este punto se analizan aquellas variables que más influyen en las predicciones del modelo para mejorar la interpretación de los resultados.

```
importances = pd.Series(rf.feature_importances_,
  index=X.columns).sort_values(ascending=False)
```

importances: contiene el nombre de las variables

pd.Series: convierte *importancias* en una *Serie de Pandas*, generando una coluna con su etiqueta.

rf.feature_importances_: contiene valores numéricos

index=X.columns: contiene los nombres de las variables

.sort_values(ascending=False): ordena las variables de menor a mayor según su nivel de importancia.

Para la presentación de los resultados de aplica:

```
print("\n Importancia de variables:")

print(importances)
```

Figura 12 Importancia de variables

```
Ventas                  0.872309
Stock_Calculado         0.102237
Criticidad              0.022842
Impacto_Monetario       0.001358
Inconsistencia_Stock    0.001100
categorias_encoded      0.000155
dtype: float64
```

La suma total de las importancias es 1 (100%), este representa el peso relativo de cada una con respecto su impacto en el modelo.

Según los resultados obtenidos, se observa que Ventas tiene un 87.23% de importancia, convirtiéndose en la variable que más influye como principal predictor de las compras, es decir que, el modelo aprende que la cantidad de compras es dependiente del comportamiento de las ventas.

Se realiza la visualización del de uno de los árboles que forman parte del bosque aleatorio, en este caso el primer árbol.

```
plt.figure(figsize=(30, 10))
```

plt.figure(): es una función de Matplotlib para crear una figura gráfica.

Figsize = (30, 10): 30 pulgadas de ancho y 1º pulgadas de alto.

```
plot_tree(rf.estimators_[0], feature_names=X.columns,
          filled=True, rounded=True, fontsize=10)
```

plot_tree (): dibuja la estructura del árbol.

rf.estimators_[0]: dentro de la lista de todos los árboles de decisión, se indica que árbol se va a graficar, en este casi se selecciona el primer árbol del bosque *[0]*.

feature_names=X.columns: esta función asigna nombres reales de las variables a los nodos para que sea interpretativo.

filled=True: les agrega color a los nodos para una mejor interpretación.

rounded=True: mejora la estética del árbol mostrando los nodos con bordes redondeados.

fontsize=10: se define el tamaño del texto que tienen los nodos.

Para el título del gráfico se aplica el siguiente código:

```
plt.title("Árbol #0 del Random Forest (Profundidad máxima =
                                    5)")
```

Para guardar el gráfico en formato png:

```
plt.savefig("random_forest_arbol_0.png", dpi=150,
                    bbox_inches='tight')
```

plt.savefig: guarda el gráfico en un archivo de imagen

"random_forest_arbol_0.png":

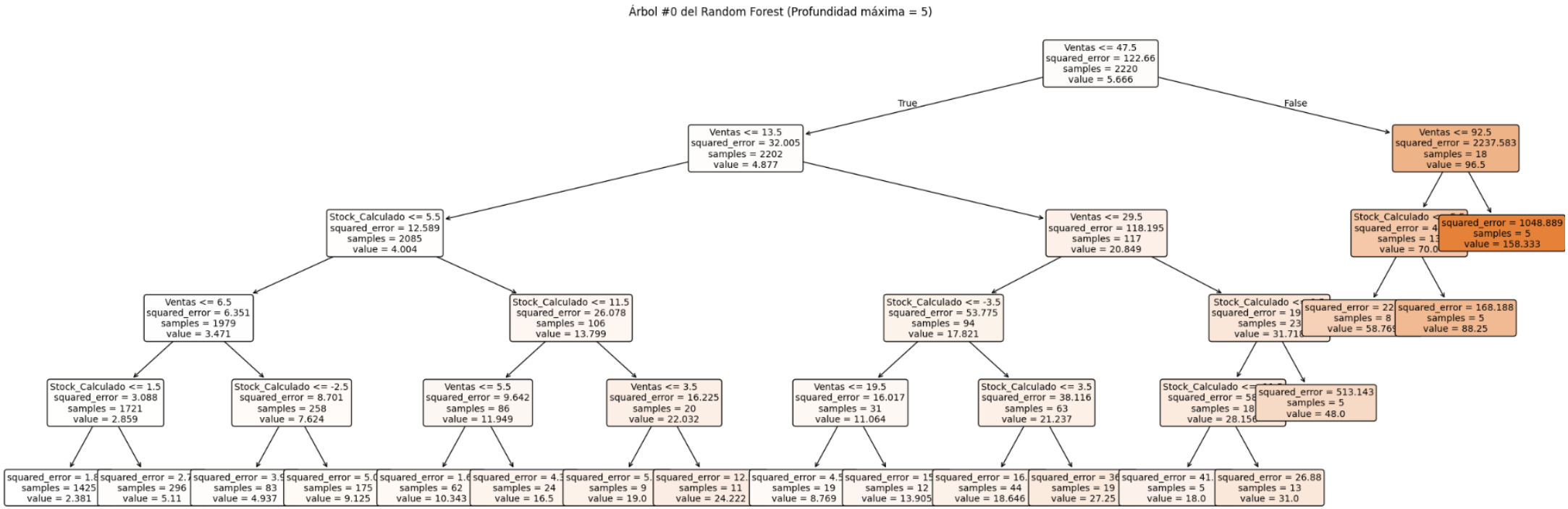
dpi=150: resolución media-alta recomendada.

bbox_inches='tight': ajusta los márgenes de la figura eliminando espacio en banco innecesarios

Finalmente, para la visualización de los datos se aplica:

```
plt.show()
```

Figura 13 Visualización del árbol #0 del modelo Random Forest Regressor



Se ponen las predicciones en todo el archivo original

```
df['Predicciones'] = rf.predict(X)
```

Figura 14 Visualización del DataFrame con la columna Predicciones generada por el modelo

	Criticidad	Cluster_KMeans	Codigo	SubGrupo	Proveedor	Marca	Descripcion	Costo_Unitario	Compras	Ventas	...	costo de venta	categorias	Stock_Calculado	Stock correcto	Diferencia	Impacto_Monetario	Rotacion	Riesgo_Stock_Negativo	Inconsistencia_Stock	Predicciones
0	Bajo	2	2349	REFRIGERANTE	PROMESA	PRESTONE	PRESTONE.	4.5450	299	256	...	1163.52	Lubricantes Filtros y refrigerantes	43	0	5	1163.52	256	0	5	159.070766
1	Bajo	2	3051	BUJIAS	L.HENRIQUEZ & CIA S.A	NGK	CHEV AVEO NISSAN B13 SWIFT STEEM SZ	1.5450	210	248	...	383.16	Sistema Eléctrico	-38	0	38	383.16	248	1	38	159.097327
2	Bajo	2	3050	BUJIAS	L.HENRIQUEZ & CIA S.A	NGK	CHEV CORSA LUV DMAX CHEVY	1.3000	190	241	...	313.30	Sistema Eléctrico	-51	0	129	313.30	241	1	129	158.648842
3	Bajo	2	3049	BUJIAS	L.HENRIQUEZ & CIA S.A	NGK	CHEV SAIL 1.4 N200 N300 BEAT SPARK GT	1.8000	150	181	...	325.80	Sistema Eléctrico	-31	0	31	325.80	181	1	31	159.097327
4	Bajo	2	2949	SILICON	PROMESA	PERMATEX	GRIS PERMATEX	2.3129	143	144	...	333.06	Siliconas y Limpiadores	-1	0	22	333.06	144	1	22	159.345349
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
4350	Alto	0	3320	CILINDRO EMBRAGUE	IMPORTADORA MAGIATY CIA. LTDA.	MGT	AUX CHEV LUV 2.5	6.6597	1	1	...	6.66	Embrague	0	1	0	6.66	1	0	0	2.176627
4351	Alto	0	2867	BOBINA ENCENDIDO	IMPORTADORA MAGIATY CIA. LTDA.	DRAPA	CHEV, ASTRA, ZAFIRA	10.5544	1	1	...	10.55	Sistema Eléctrico	0	1	0	10.55	1	0	0	2.176627
4352	Alto	0	2868	BOBINA ENCENDIDO	IMPORTADORA MAGIATY CIA. LTDA.	DRAPA	SUZ. S-CROSS, VITARA SZ.	10.5541	4	1	...	10.55	Sistema Eléctrico	3	1	0	10.55	1	0	0	4.573589
4353	Alto	0	2871	PUNTA INT	IMPORTADORA MAGIATY CIA. LTDA.	GMSGS	CHEV SAIL 1.4	19.2000	4	1	...	19.20	Direccion y Suspension	3	1	0	19.20	1	0	0	4.573589
4354	Alto	0	2309	EMPAQUE TAPA VALVULA	IMPORTADORA CERON	ONNURI	HY EXCEL.	1.1300	1	1	...	1.13	Repuestos de motor	0	1	0	1.13	1	0	0	2.176627

4355 rows × 22 columns

Discusión

Se examinó el artículo científico "*Ad-Campaign Analysis and Sales Prediction using K-means Clustering and Random Forest Regressor*" de Ghosh junto con Gor, publicado en el IOSR Journal of Mathematics (2022), con el propósito de comparar la metodología aplicada en el presenta trabajo de titulación "*Aplicación del Machine Learning para la clasificación, predicción e importación de repuestos críticos para automotores*". Esta investigación utiliza un método que integra K-means para los clusters (o agrupamiento) y Random Forest Regressor para las predicciones, lo cual es consistente desde el punto de vista metodológico con el enfoque que se ha desarrollado en la presente tesis.

El propósito principal del artículo fue examinar las campañas de publicidad digital y pronosticar las ventas futuras utilizando métodos de aprendizaje automático. La situación planteada radica en detectar patrones en los datos relacionados con la publicidad (gasto, impresiones y clics) con el fin de mejorar la conversión de ventas.

A pesar de que el artículo se centre en marketing digital y el presente estudio se oriente al sector automotriz para clasificar, predecir e importar repuestos críticos, ambos estudios comparten un enfoque metodológico similar:

1. Segmentación preliminar a través de aprendizaje no supervisado (K-means).

2. Predicción de una variable continua a través del aprendizaje supervisado, usando el Random Forest Regressor.

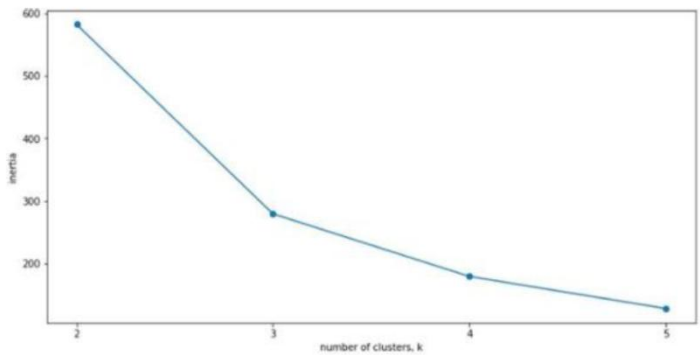
Esta similitud de los métodos aplicados posibilita a establecer una comparación técnica. El artículo parte con el desarrollo del K-means, los autores emplean este algoritmo de cluster

en la fase inicial para segmentar los anuncios publicitarios en diferentes segmentos basándose en tres variables independientes:

1. Impresiones (frecuencia con la que se presenta el anuncio)
2. Clicks (cantidad de clics obtenidos)
3. Gasto (cantidad de dinero invertida en publicidad)

Utilizan el método del codo (Elbow Method) para calcular la suma de cuadrados dentro del grupo (WCSS, sus siglas en inglés) y de esta manera establecer el número ideal de clusters (K). La gráfica del método del codo (Graph 1 en el artículo) revela que el valor ideal elegido fue K = 3, lo que señala la presencia de tres segmentos distintos de campañas publicitarias.

Figura 15 Método del codo para calcular k



Graph 1: Elbow method to calculate K

Posteriormente, los autores representan gráficamente la correlación entre las variables tenidas en cuenta en el estudio: clics, impresiones y gasto publicitario. Para esto, crean tres gráficos de dispersión en los que cada punto simboliza una campaña publicitaria y cada color señala el grupo al que esa campaña pertenece, de acuerdo con la segmentación hecha por el algoritmo.

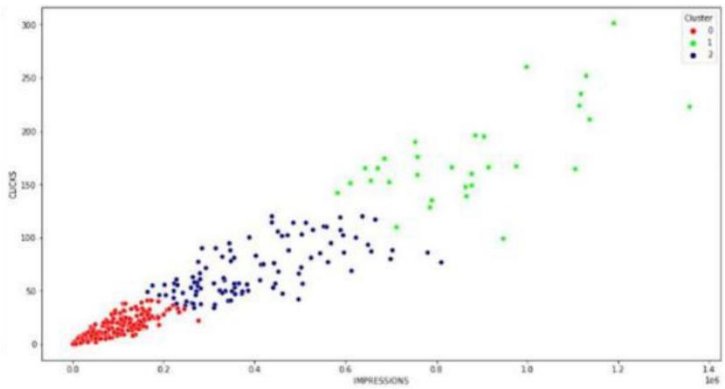
- Graph 2: Relación entre impresiones y clics

- Graph 3: Relación entre impresiones y gasto
- Graph 4: Relación entre gasto y clics

En estos gráficos, se pueden distinguir tres grupos de colores diferentes (verde, azul y rojo), los cuales representan:

- Grupo 1: Nivel bajo de clics, impresiones e inversión.
- Grupo 2: Rango medio de inversión y rendimiento.
- Grupo 3: Clics, impresiones y una gran inversión.

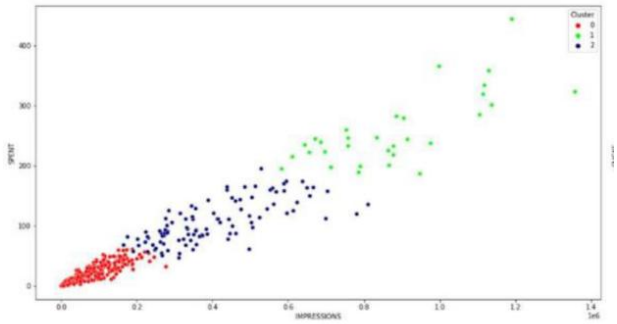
Figura 16 Relación entre impresión y clic



raph 2: Relationship between impression and click

El Gráfico 2, se analiza la correlación entre los clics y las impresiones. Este gráfico muestra cómo el total de clics que se reciben se ve afectado por la cantidad de veces que un anuncio es mostrado (impresiones). A la vista, se pueden distinguir tres agrupaciones que son claramente diferentes entre sí. El primer cluster se refiere a campañas con niveles bajos de impresiones y, por lo tanto, con pocos clics. Las campañas que tienen un nivel medio de clics e impresiones se clasifican en el segundo cluster. Por último, el tercer cluster reúne campañas con una exposición elevada y un número de clics mucho mayor. Esta distribución muestra una correlación positiva entre las dos variables: cuantas más impresiones, más clics se generan.

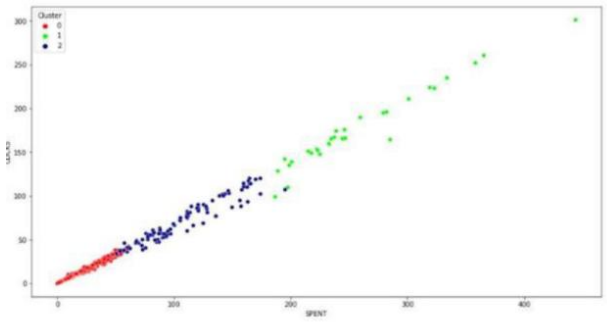
Figura 17 Relación entre impresión y gasto



Graph 3: Relationship between impression and spent

El Gráfico 3, analiza la correlación entre el gasto en publicidad y las impresiones. Se puede observar aquí que las campañas con más gastos suelen producir una cantidad mayor de impresiones. Esto resulta relevante desde un punto de vista operativo, puesto que, una inversión más alta permite que el anuncio se muestre con mayor frecuencia en la plataforma digital. Los tres clústeres, de manera similar, señalan distintos niveles de inversión: campañas con presupuesto bajo, medio y alto. La segmentación ratifica que el gasto constituye un factor crucial para determinar la cobertura del anuncio.

Figura 18 Relación entre gasto y clic



Graph 4: Relationship between spent and click

El Gráfico 4, por su parte, estudia la correlación entre el gasto y los clics. En este caso, se evidencia que las campañas con menor inversión producen menos clics, mientras que las

que tienen una mayor inversión tienden a lograr una mayor interacción de los usuarios. Esto indica que el gasto afecta no solo la cantidad de impresiones, sino también de manera indirecta la tasa de interacción del anuncio.

En conjunto, el análisis de los tres gráficos permite llegar a la conclusión de que las variables clics, gasto e impresiones están claramente interrelacionadas. El gasto en publicidad tiene impacto sobre el número de impresiones, mientras que, las impresiones tienen un impacto en la cantidad de clics. Por lo tanto, estas variables están correlacionadas, esta correlación valida que los escritores las utilicen como variables independientes en el modelo de regresión con el fin de pronosticar la conversión total. Es decir, el clustering no solo divide las campañas, sino que además permite comprender la estructura interna de los datos antes de realizar las predicciones.

Desde un punto de vista metodológico, estos gráficos desempeñan un papel fundamental, como lo es validar visualmente que los clusters creados por el K-means tengan sentido y lógica, puesto que esta clasificación refleja los niveles de desempeño y estrategia publicitaria.

Por consiguiente, una vez aplicado el algoritmo K-means, inicia la segunda etapa aplicando Random Forest Regressor para predecir la variable dependiente *Total Conversion*: número total de personas que solicitaron información sobre el producto después de ver el anuncio.

Para el entrenamiento se ocupó:

- 80% de los datos para el entrenamiento
- 20% de los datos para la prueba

Las métricas de evaluación que se utilizaron fue el MAE (error absoluto medio) para calcular el error entre las ventas futuras reales y las predichas.

Como se observa en la tabla 1, se determinó que la precisión del modelo es de 75,3 %, dado que el valor R^2 es 0,753 y el error absoluto medio es 0,991.

Figura 19 Tabla del Error Absoluto Medio y R2

Table:1 Mean Absolute Error and R-squared

Random Forest Regressor	
Mean Absolute Error	0.991
R-squared value	0.753

Los resultados indican que, según el coeficiente de determinación (R^2) el modelo explica el 75.3% de la variabilidad de los datos, lo que representa un nivel aceptable de precisión predictiva.

La metodología empleada en esta investigación confirma la validez del estudio metodológico del artículo "Ad-Campaign Analysis and Sales Prediction using K-means Clustering and Random Forest Regressor". El complemento del K-means para agrupar previamente al aplicar Random Forest Regressor es parte de una metodología sólida que se ha aplicado para resolver problemas cuando:

- Los datos contienen patrones que no son evidentes.
- Es necesaria una clasificación previa.
- Se busca predecir un valor cuantitativo.

Pese a que varía el enfoque en cada estudio, la estructura metodológica es transferible y consistente, lo que respalda la selección de técnicas para el presente trabajo de titulación.

En conclusión, el estudio examinado proporciona un fundamento tanto práctico como teórico para validar la aplicación de ambos métodos, uno de aprendizaje supervisado y otro no supervisado en cuestiones de optimización y pronóstico en el contexto de las autopartes del sector automotriz.

Conclusiones

Con respecto al primer objetivo, la revisión de la literatura científica reveló que el aprendizaje automático se ha consolidado como una herramienta estratégica para la gestión de inventarios y la toma de decisiones en entornos empresariales. Los documentos obtenidos demostraron la aplicabilidad de los métodos de clasificación y predicción a diversos escenarios productivos y comerciales, particularmente utilizando información histórica confiable. Esta revisión teórica no solo validó el uso de modelos predictivos en el sector automotriz, sino que también apoyó en la selección de métodos adecuados para el tratamiento y análisis de datos sobre piezas de repuesto críticas.

Con respecto al segundo objetivo, la construcción y aplicación del modelo de clasificación y predicción proporcionó evidencia de que la toma de decisiones en la importación de repuestos puede mejorarse al tener en cuenta la rotación, el impacto monetario, el comportamiento de ventas, los niveles de stock y otras variables. El modelo podría ser valioso para reconocer patrones que pueden no ser fácilmente evidentes, evitando así el exceso de inventario o la falta de existencias. Por lo tanto, está claro que el uso de métodos de aprendizaje automático específicamente en el problema de adquisición fue una forma exitosa de mitigar cualquier desventaja de los métodos de aplicación clásicos que se basan únicamente en la experiencia empírica o el juicio subjetivo.

La interpretación en relación con el tercer objetivo se proporcionó a través de la discusión de resultados en este informe, lo cual se justificó para la interpretación del rendimiento del algoritmo subyacente y la comparación con los estudios relacionados en la literatura. Los resultados observados sugieren que esta metodología juega un papel sustancial en la optimización de la gestión de repuestos al permitir una mejor clasificación de las piezas críticas y mejorar la planificación de importaciones. Lo mismo ocurrió con el uso de la tecnología en el ámbito automotriz, donde la aplicación de herramientas en este contexto no

solo mejora la eficiencia operativa, sino que también añade valor estratégico a la toma de decisiones organizacionales.

En general, este estudio confirma que el aprendizaje automático para la clasificación y pronóstico de partes críticas es una alternativa práctica, relevante y adaptable a las condiciones del sector automotriz. Además, la adopción de modelos basados en datos facilita la reducción de la incertidumbre, optimiza la gestión de inventarios y apoya el fortalecimiento competitivo de las empresas involucradas en la venta e importación de autopartes.

Bibliografías

Aco Tito, A. E., Hancco Condori, B. O., & Pérez Vera, Y. (septiembre de 2023). Análisis comparativo de Técnicas de Machine Learning para la predicción de casos de deserción universitaria. Revista lbérica de Sistemas y Tecnologías de Información(51), 84-98. doi:10.17013/risti.51.84–98

Barragán-Martínez, X. (2023). Situación de la Inteligencia Artificial en el Ecuador en relación con los países líderes de la región del Cono Sur. FIGEMPA: Investigación y Desarrollo, 16(2), 23-38. doi:https://doi.org/10.29166/revfig.v16i2.4498

Brodie, Michael L. (2019). What Is Data Science? . En M. S. Braschler, & T. S. Martin Braschler (Ed.), Applied Data Science (págs. 101-130). Springer Cham. doi:https://doi.org/10.1007/978-3-030-11821-1

Caiafa, C. F., & Lew, S. E. (17 de 06 de 2020). ¿Qué es la Inteligencia Artificial? Boletín Radio@stronómico, 1-7. Obtenido de https://www.iar.unlp.edu.ar/boletin/que-es-la-inteligencia-artificial/

Calle García, J. S., Pincay Delgado, M. A., Mendoza Pionce, B. S., & Bravo Quijije, G. S. (2024). Uso estratégico de la inteligencia artificial en la gestión de la cadena de suministro empresarial. Ciencia y Desarrollo, 27(2), 268–275. Obtenido de http://revistas.uap.edu.pe/ojs/index.php/CYD/index

Carlos, O. S., Manuel, M. A., & Eduardo, O. P. (junio de 2023). Análisis multivariante. Regresión lineal múltiple. Evidencias en Pediatría , 10(2), 1-7. Obtenido de https://evidenciasenpediatria.es/files/41-14389-RUTA/EeP_Fund_22_AnalisisMultivariante_RegLinealMultiple.pdf

Chahboun, S., & Maarouf, M. (2021). Principal Component Analysis and Machine Learning Approaches for Photovoltaic Power Prediction: A Comparative Study. Applied Sciences(17), 11. doi:https://doi.org/10.3390/app11177943

Chahboun, S., & Maarouf, M. (2021). Principal Component Analysis and Machine Learning Approaches for Photovoltaic Power Prediction: A Comparative Study. Applied Sciences(17), 11. doi:https://doi.org/10.3390/app11177943

Challenger-Pérez, I., Díaz-Ricardo, Y., & Becerra-García, R. A. (abril-junio de 2014). El lenguaje de programación Python. Ciencias Holguín, XX(2), 1-13. Obtenido de https://www.redalyc.org/pdf/1815/181531232001.pdf

Chopra, S., & Meindl, P. (2019). Supply chain management: Strategy, planning, and operation (7th ed.). Pearson. https://www.pearson.com/en-us/subject-catalog/p/supply-chain-management-strategy-planning-and-operation/P200000003479

Contretas Bravo, L. E., & Padilla Beltrám, J. E. (2024). Ciencia de datos con Python. Transformación y selección de variables (Primera ed.). Bogotá, Colombia : Ediciones de la U. Obtenido de https://content.e-bookshelf.de/media/reading/L-25736721-eaba73c9c0.pdf

Dagnino S., J. (2024). Regresión Lineal. Revista Chilena de Anestesia , 43(2), 143-149. doi:https://doi.org/10.25237/revchilanestv43n02.14

Durán, Y. (enero-junio de 2012). Administración del inventario: elemento clave para la optimización de las utilidades en las empresas . (1), 55-78. Obtenido de https://www.redalyc.org/pdf/4655/465545892008.pdf

Feizabadi, J. (2022). Machine learning demand forecasting and supply chainperformance. International Journal of Logistics Research and Applications, 25(2), 119-142. doi:https://doi.org/10.1080/13675567.2020.1803246

Font, X. (2019). Técnicas de clasificación supervised learning. Barcelona: Oberta UOC Publishing, SL. Obtenido de https://openaccess.uoc.edu/server/api/core/bitstreams/25c73efb-0646-4811-83f1-aa1eb3772896/content

Font, X. (2019). Técnicas de clasificación supervised learning. Barcelona: Oberta UOC Publishing, SL. Obtenido de https://openaccess.uoc.edu/server/api/core/bitstreams/25c73efb-0646-4811-83f1-aa1eb3772896/content

Font, X. (2019). Técnicas de clustering. Barcelona: FUOC/Oberta UOC Publishing. Obtenido de https://openaccess.uoc.edu/server/api/core/bitstreams/859ca353-d4f7-4448-a284-6454decfc950/content

Ghosh, M., & Ravi, G. (2022). Ad-Campaign Analysis and Sales prediction using Kmeans Clustering andRandom Forest Regressor. IOSR Journal of Mathematics, 18(2), 05-09. doi:10.9790/5728-1802020509

González Duque, R. (2007). Python para todos. Obtenido de https://persoal.citius.usc.es/eva.cernadas/informaticaparacientificos/material/libros/Python%20para%20todos.pdf

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT Press. https://www.deeplearningbook.org/

Harris, C. R., Millman, K. J., van der Walt, S. J., et al. (2020). Array programming with NumPy. Nature, 585(7825), 357–362. https://www.nature.com/articles/s41586-020-2649-2

Hernández, H. A., Cruz-Gil, Y. L., Puentes Saavedra, M. D., y Mendoza Patiño, D. E. (2021). Diseño de un sistema de gestión de inventarios para el almacén Técnitaller S.A.S de la ciudad Neiva-huila, Colombia. Revista de Investigaciones Universidad Del Quindío, 33(2), 143–152. https://doi.org/10.33975/riuq.vol33n2.562

Hosmer, C. (2014). Forensics, Python. (C. Hosmer, Ed.) 1-11. doi:https://doi.org/10.1016/B978-0-12-418676-7.00001-3

Iñiguez Pineda, M. J., & Maldonado Ruiz, L. M. (abril-junio de 2025). Protección de datos personales frente al desarrollo de inteligencia artificial en Ecuador: análisis y propuestas legislativas. Revista en Ciencias sociales y humanidades, 9(2), 24-48. doi:https://doi.org/10.33262/visionariodigital.v9i2.3366

Jara Obregón, L. S., & Naspud Espinoza, M. G. (2025). Inteligencia Artificial: Desafíos y Oportunidades Para Las Pymes Ecuatorianas. Arandu UTIC, 11(2), 3063–3077. doi:https://doi.org/10.69639/arandu.v11i2.485

Kagalwala, H., Radhakrishnan, G. V., Mohammed, I. A., Kothinti, R. R., & Kulkarni, N. (2025). Predictive Analytics in Supply Chain Management: The Role of AI and

Machine Learning in Demand Forecasting. Advances in Consumer Research, 2(1), 142-149. Obtenido de file:///C:/Users/Usuario/Downloads/ACR-12..pdf

Lemenkova, P. (2020). Python Libraries Matplotlib, Seaborn and Pandas for Visualization Geo-spatial Datasets Generated by QGIS Analele stiintifice ale Universitatii "Alexandru Ioan Cuza" din Iasi - seria Geografie, vol. 64(1), pp. 13-32, 2020, Available at SSRN: https://ssrn.com/abstract=3699706

León González, Á., Llinás Solano, H., & Tilano, J. (23 de enero-junio de 2008). Análisis multivariado aplicando componentes principales al caso de los desplazados. Ingeniería y Desarrollo(23), 119-142. Obtenido de https://www.redalyc.org/pdf/852/85202310.pdf

Liu, Y., Kalaitzi, D., Wang, M., & Papanagnou, C. (2025). A machine learning approach to inventory stockout prediction. Journal of Digital Economy, 4, 144–155. https://doi.org/10.1016/j.jdec.2025.06.002

López de Mántaras, R., & Brunet, P. (2023). Riesgos, ventajas y repercusiones. de la Inteligencia Artificial. PAPELES de relaciones ecosociales y cambio global(164), 13-21. Obtenido de https://www.fuhem.es/papeles_articulo/que-es-la-inteligencia-artificial/

Martínez Pérez, J., & Pérez Martin, P. (enero-febrero de 2023). La curva ROC. Medicina en Familia. SEMERGEN, 49(1). doi:10.1016/j.semerg.2022.101821

McKinney, W. (2010). Data structures for statistical computing in Python. Proceedings of the 9th Python in Science Conference, 51–56. https://doi.org/10.25080/Majora-92bf1922-00a

Mendoza, J., Macías, L., Morales, J., & Cedeño, L. (2024). Modelos de Aprendizaje Automático: Aplicación y Eficiencia . Revista Científica de Informática ENCRIPTAR, 7(14). doi:https://doi.org/10.56124/encriptar.v7i14.005

Moral Peláez, I. (2016). Modelos de regresión: lineal simple y regresión logística . Revista Seden , 14, 195-214. Obtenido de https://www.revistaseden.org/files/14-cap%2014.pdf

Parra Angel, S., & Fuentes Rojas, E. Á. (2023). Desarrollo de un sistema de gestión de inventarios para el control de materiales, equipos y herramientas dentro de la empresa de construcción Realidad Colombia S.A.S. Revista Ingeniería, Matemáticas y Ciencias de la Información, 10(19), 61–72. https://ojs.urepublicana.edu.co/index.php/ingenieria/article/view/878/630

Poveda-Valverde, & Flor. (14 de julio de 2025). AI Applications That Can Support Sustainable Practices in Small and Medium-Sized Enterprises in Latin America: A Systematic Review. Preprints.org, 12. doi:10.20944/preprints202507.0880.v1

Rainio, O., Teuho, J., & Riku, K. (2024). Evaluation metrics and statistical tests for machine learning. Scientific Reports(6068). doi:https://doi.org/10.1038/s41598-024-56706-x

Ramírez, W., Antúnez, G., & Rodríguez, Y. (marzo de 2016). La utilización del Análisis de los Componentes Principales en la Medicina Veterinaria. REDVET. Revista Electrónica de Veterinaria, 17(3), 1-8. Obtenido de https://www.redalyc.org/pdf/636/63646040001.pdf

Rodríguez, A., Sabogal Cáceres, T., & Fuentes Rojas, E. (2021). Sistema de gestión de inventarios para compañías de hardware – Caso de estudio. Revista Ingeniería,

Matemáticas y Ciencias, 8(16), 27–36 https://ojs.urepublicana.edu.co/index.php/ingenieria/article/view/748/560

Ruiz Aranibar, G. (noviembre de 2018). Estadística multivariable análisis de componentes principales. REVISTA VARIANZA(15), 44-61. Obtenido de https://ojs.umsa.bo/index.php/revistavarianza/article/view/399

Russell, S., & Norvig, P. (2021). Artificial intelligence: A modern approach (4th ed.). Pearson. https://aima.cs.berkeley.edu/newchap00.pdf Bishop, C. M. (2006). Pattern recognition and machine learning. Springer. https://www.microsoft.com/en-us/research/wp-content/uploads/2006/01/Bishop-Pattern-Recognition-and-Machine-Learning-2006.pdf

Sanchez Ruiz, L., Blanco, B., & Kyguoliené, A. (2018). A Theoretical Overview of the Stockout Problem in Retail: from Causes to Consequences. Sciendo . doi:https://doi.org/10.1515/mosr-2018-0007

Soto Bravo, F., & González Lutz, M. I. (2019). Análisis de métodos estadísticos para evaluar el desempeño de modelos de simulación en cultivos hortícolas. Agronomía Mesoamericana, 30(2), 517-534. doi:10.15517/am.v30i2.33839

Terven, J., Cordova Esparza, D. M., Romero González, J. A., Pedraza, A. R., & Chávez-Urbiola, E. A. (11 de Abril de 2025). A comprehensive survey of loss functions and metrics in deep learning. 58(195). doi:https://doi.org/10.1007/s10462-025-11198-7

Valencia, H. C. (enero de 2025). Agentes de software basados en técnicas de aprendizaje automático. Perspectivas desde 2010 hasta 2023. Revista Colombiana de Tecnologías de Avanzada, 1(45), 39-56. Obtenido de https://dialnet.unirioja.es/servlet/articulo?codigo=9989430

Vergara Villegas, O. O. (28 de febrero de 2021). Artificial Intelligence for Industry 4.0 in Iberoamerica. Computación y Sistemas. Computación y Sistemas, 25(4), 761-773. doi:https://doi.org/10.13053/cys-25-4-4056

Verma, P., & Gupta, R. K. (junio de 2018). A Literature Survey on Classification Algorithms of Machine Learning. International Journal of Computer Applications, 179(53), 47-50. doi:https://doi.org/10.5120/ijca2018917378

Vilà Baños, R., Torrado Fonseca, M., & Reguant Álvarez, M. (julio de 2019). Análisis de regresión lineal múltiple con SPSS: un ejemplo práctico. REIRE Revista d'Innovació i Recerca en Educació, 12(2), 34-50. doi:https://doi.org/10.1344/reire2019.12.222704

Waskom, M. L. (2021). seaborn: Statistical data visualization. Journal of Open Source Software, 6(60), 3021. https://joss.theoj.org/papers/10.21105/joss.03021

Anexos

Script del Modelo en Python

CARGA DE DATOS

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
#Cargamos los datos desde la pc
from google.colab import files
uploaded = files.upload()
```

```
Elegir archivos  Sin archivos seleccionados  Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving Michela y Renata.xlsx to Michela y Renata.xlsx
```

```
#Leemos los datos de excel
data = pd.read_excel('Michela y Renata.xlsx')
```

```
data
```

Mostrar salida oculta

LIMPIEZA DE DATOS

```
#Corregir nombre de columna mal escrito
data.rename(columns={"Stock Calcualdo": "Stock_Calculado"}, inplace=True)
```

```
#Función para convertir strings con comas a float
def clean_numeric(x):
    if pd.isna(x):
        return np.nan
    if isinstance(x, str):
        x = x.replace(',', '')
    try:
        return float(x)
    except ValueError:
        return np.nan
```

```
# Aplicar limpieza a columnas numéricas problemáticas
numeric_cols = [
    'Costo_Unitario', 'costo compra', 'costo de venta'
]
for col in numeric_cols:
    data[col] = data[col].apply(clean_numeric)
```

```
#Asegurar tipos numéricos en columnas enteras
int_cols = ['Compras', 'Ventas', 'Stock', 'Stock_Calculado', 'Stock correcto', 'Diferencia']
for col in int_cols:
    data[col] = pd.to_numeric(data[col], errors='coerce').fillna(0).astype(int)
```

```
#Validar o recalcular Stock_Calculado (opcional: usar el dato existente o recalcular)
# Aquí optamos por confiar en el cálculo lógico: Compras - Ventas
data['Stock_Calculado'] = data['Compras'] - data['Ventas']
```

+ Código + Texto

```
# Crear variables derivadas útiles para criticidad
data['Impacto_Monetario'] = data['costo de venta']  # o abs(df['costo de venta'])
data['Rotacion'] = data['Ventas']
data['Riesgo_Stock_Negativo'] = (data['Stock_Calculado'] < 0).astype(int)
data['Inconsistencia_Stock'] = np.abs(data['Stock'] - data['Stock_Calculado'])
```

```
#Eliminar filas completamente vacías
data.dropna(how='all', inplace=True)
```

```
#Eliminar duplicados exactos (por Código o por todas las columnas)
data.drop_duplicates(subset=['Codigo'], keep='first', inplace=True)
```

```
#Seleccionar solo las columnas relevantes para clustering
features_for_clustering = [
    'Rotacion',
    'Impacto_Monetario',
    'Stock_Calculado',
    'Inconsistencia_Stock'
]

X = data[features_for_clustering].copy()
```

```
#Manejo de NaN en variables derivadas (poco probable, pero seguro)
X.fillna(0, inplace=True)
```

```
#Guardar datos limpios (opcional)
data.to_excel("Michela_y_Renata_limpio.xlsx", index=False)
X.to_csv("datos_para_kmeans.csv", index=False)

print("✅ Limpieza completada.")
print("📊 Forma del conjunto para clustering:", X.shape)
print("\nPrimeras filas:")
print(X.head())
```

```
✅ Limpieza completada.
📊 Forma del conjunto para clustering: (4355, 4)

Primeras filas:
   Rotacion  Impacto_Monetario  Stock_Calculado  Inconsistencia_Stock
0       256            1163.52               43                     5
1       248             383.16              -38                    38
2       241             313.30              -51                   129
3       181             325.80              -31                    31
4       144             333.06               -1                    22
```

```
#Descargar los datos limpios a la pc
files.download("Michela_y_Renata_limpio.xlsx")
```

APLICAMOS K-MEANS (CLASIFICACIÓN DE CRÍTICOS Y NO CRÍTICOS)

```
#Cargamos las librerías necesarias
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.metrics import silhouette_score, calinski_harabasz_score
```

```
#Cargamos los datos desde la pc
from google.colab import files
uploaded = files.upload()
```

```
Elegir archivos  Sin archivos seleccionados  Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving Michela_y_Renata_limpio.xlsx to Michela_y_Renata_limpio.xlsx
```

```
#Leemos los datos limpios
df = pd.read_excel("Michela_y_Renata_limpio.xlsx", sheet_name="Sheet1")
```

```
df
```

	Codigo	SubGrupo	Proveedor	Marca	Descripcion	Costo_Unitario	Compras	Ventas	Stock	costo compra	costo de venta	categorias	Stock_Calculado	Stock correcto	Diferencia	Impacto_Monetario	Rotacion	Riesgo_Stock_Negativo	Inconsistencia_
0	2349	REFRIGERANTE	PROMESA	PRESTONE	PRESTONE.	4.5450	299	256	48	1358.96	1163.52	Lubricantes Filtros y refrigerantes	43	0	5	1163.52	256	0	
1	3051	BUJIAS	L.HENRIQUEZ & CIA S.A	NGK	CHEV AVEO NISSAN B13 SWIFT STEEM SZ	1.5450	210	248	0	324.45	383.16	Sistema Eléctrico	-38	0	38	383.16	248	1	
2	3050	BUJIAS	L.HENRIQUEZ & CIA S.A	NGK	CHEV CORSA LUV DMAX CHEVY	1.3000	190	241	78	247.00	313.30	Sistema Eléctrico	-51	0	129	313.30	241	1	
3	3049	BUJIAS	L.HENRIQUEZ & CIA S.A	NGK	CHEV SAIL 1.4 N200 N300 BEAT SPARK GT	1.8000	150	181	0	270.00	325.80	Sistema Eléctrico	-31	0	31	325.80	181	1	
4	2949	SILICON	PROMESA	PERMATEX	GRIS PERMATEX	2.3129	143	144	21	330.74	333.06	Siliconas y Limpiadores	-1	0	22	333.06	144	1	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
350	3320	CILINDRO EMBRAGUE	IMPORTADORA MAGIATY CIA. LTDA.	MGT	AUX CHEV LUV 2.5	6.6597	1	1	0	6.66	6.66	Embrague	0	1	0	6.66	1	0	
351	2867	BOBINA ENCENDIDO	IMPORTADORA MAGIATY CIA. LTDA.	DRAPA	CHEV, ASTRA, ZAFIRA	10.5544	1	1	0	10.55	10.55	Sistema Eléctrico	0	1	0	10.55	1	0	
352	2868	BOBINA ENCENDIDO	IMPORTADORA MAGIATY CIA. LTDA.	DRAPA	SUZ. S-CROSS, VITARA SZ.	10.5541	4	1	3	42.22	10.55	Sistema Eléctrico	3	1	0	10.55	1	0	
353	2871	PUNTA INT	IMPORTADORA MAGIATY CIA.	GMSGS	CHEV SAIL 1.4	19.2000	4	1	3	76.80	19.20	Direccion y Suspension	3	1	0	19.20	1	0	
354	2309	EMPAQUE TAPA VALVULA	IMPORTADORA CERON	ONNURI	HY EXCEL.	1.1300	1	1	0	1.13	1.13	Repuestos de motor	0	1	0	1.13	1	0	

55 rows × 19 columns

```python
#Seleccionar variables para clustering
features = ['Rotacion', 'Impacto_Monetario', 'Stock_Calculado', 'Inconsistencia_Stock']
X = df[features].copy()
```

```python
#Estandarización (K-means es sensible a escalas)
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

```python
#Aplicar K-means con k=3
kmeans = KMeans(n_clusters=3, random_state=42, n_init=10)
df['Cluster_KMeans'] = kmeans.fit_predict(X_scaled)
```

```python
#Calcular el centroide promedio de cada cluster (en escala original para interpretación)
centroids_scaled = kmeans.cluster_centers_
centroids_original = scaler.inverse_transform(centroids_scaled)
centroid_df = pd.DataFrame(centroids_original, columns=features)
```

```python
# 7. Asignar etiquetas: "Alto", "Medio", "Bajo" según rotación e impacto
# Ordenar clusters por rotación descendente
centroid_df['cluster_id'] = centroid_df.index
centroid_df_sorted = centroid_df.sort_values(by='Rotacion', ascending=False).reset_index(drop=True)
```

```python
# Mapeo: el primero = Alto, segundo = Medio, tercero = Bajo
cluster_stats = df.groupby('Cluster_KMeans')[['Rotacion', 'Impacto_Monetario']].mean()
critical_order = cluster_stats.sort_values(by='Rotacion', ascending=False).index
label_map = {centroid_df_sorted.loc[0, 'cluster_id']: 'Bajo',
             centroid_df_sorted.loc[1, 'cluster_id']: 'Medio',
             centroid_df_sorted.loc[2, 'cluster_id']: 'Alto'}

df['Criticidad'] = df['Cluster_KMeans'].map(label_map)
```

```python
# Métricas de validación
sil_score = silhouette_score(X_scaled, df['Cluster_KMeans'])
ch_score = calinski_harabasz_score(X_scaled, df['Cluster_KMeans'])
```

```python
#Reordenar columnas
cols = list(df.columns)
cols.insert(0, cols.pop(cols.index('Criticidad')))
cols.insert(1, cols.pop(cols.index('Cluster_KMeans')))
df = df[cols]
```

```python
#Resumen
summary = df['Criticidad'].value_counts().reindex(['Bajo', 'Medio', 'Alto'], fill_value=0)
print(f"✅ Clustering completado con k=3.")
print(f"📊 Distribución de criticidad:")
for level, count in summary.items():
    print(f"   • {level}: {count} ítems")
```

```
✅ Clustering completado con k=3.
📊 Distribución de criticidad:
   • Bajo: 6 ítems
   • Medio: 240 ítems
   • Alto: 4109 ítems
```

```python
# Reducción a 2D para visualización
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_scaled)
```

```python
# --- Gráfico 1: Proyección PCA ---
plt.figure(figsize=(8, 6))
colors = {'Bajo': 'green', 'Medio': 'orange', 'Alto': 'red'}  # ¡Corregido!

for level in ['Bajo', 'Medio', 'Alto']:
    mask = df['Criticidad'] == level
    plt.scatter(
        X_pca[mask, 0],
        X_pca[mask, 1],
        c=colors[level],     # <-- ¡Asignación explícita de color!
        label=level,
        alpha=0.6,
        s=20
    )

plt.title('K-means Clusters (Proyección PCA)')
plt.xlabel(f'PC1 ({pca.explained_variance_ratio_[0]:.1%} varianza)')
plt.ylabel(f'PC2 ({pca.explained_variance_ratio_[1]:.1%} varianza)')
plt.legend()
plt.tight_layout()
plt.savefig("kmeans_pca.png", dpi=150)
plt.show()
```

Mostrar salida oculta

```python
# --- Gráfico 2: Perfiles normalizados de clusters ---
profile = df.groupby('Criticidad')[features].mean()
profile_norm = (profile - profile.min()) / (profile.max() - profile.min())

fig, ax = plt.subplots(figsize=(8, 5))
x = np.arange(len(features))
width = 0.25
ax.bar(x - width, profile_norm.loc['Bajo'], width, label='Bajo', color='green', alpha=0.7)
ax.bar(x,         profile_norm.loc['Medio'], width, label='Medio', color='orange', alpha=0.7)
ax.bar(x + width, profile_norm.loc['Alto'], width, label='Alto', color='red', alpha=0.7)
ax.set_xticks(x)
ax.set_xticklabels(features, rotation=45, ha='right')
ax.set_ylabel('Valor normalizado (0-1)')
ax.set_title('Perfiles Normalizados por Nivel de Criticidad')
ax.legend()
plt.tight_layout()
plt.savefig("cluster_profiles.png", dpi=150)
plt.show()
```

Mostrar salida oculta

```python
# --- Gráfico 3: Conteo de ítems por cluster ---
counts = df['Criticidad'].value_counts().reindex(['Bajo', 'Medio', 'Alto'])
plt.figure(figsize=(6, 4))
bars = plt.bar(counts.index, counts.values, color=['red', 'orange', 'green'], alpha=0.7)
for bar in bars:
    plt.text(bar.get_x() + bar.get_width()/2, bar.get_height() + 20,
             str(int(bar.get_height())), ha='center')
plt.title('Cantidad de Ítems por Nivel de Criticidad')
plt.ylabel('Número de ítems')
plt.tight_layout()
plt.savefig("cluster_counts.png", dpi=150)
plt.show()
```

Mostrar salida oculta

```python
# --- Gráfico 4: Boxplots por característica ---
fig, axes = plt.subplots(2, 2, figsize=(12, 8))
axes = axes.ravel()
for i, col in enumerate(features):
    data = [df[df['Criticidad'] == lvl][col].values for lvl in ['Alto', 'Medio', 'Bajo']]
    bp = axes[i].boxplot(data, tick_labels=['Bajo', 'Medio', 'Alto'], patch_artist=True)

    # Set colors for each box manually
    for patch, color_name in zip(bp['boxes'], ['Bajo', 'Medio', 'Alto']):
        patch.set_facecolor(colors[color_name])

    axes[i].set_title(f'{col}')
    axes[i].tick_params(axis='x', rotation=45)
plt.suptitle('Distribución de Variables por Nivel de Criticidad')
plt.tight_layout()
plt.savefig("boxplots_features.png", dpi=150)
plt.show()
```

Mostrar salida oculta

```python
#Guardar resultado
output_file = "Michela_y_Renata_con_Criticidad.xlsx"
df.to_excel(output_file, index=False)
```

```python
#Descargamos a la pc
files.download(output_file)
```

...

+ Código + Texto

APLICAMOS RANDOM FOREST PARA LA PREDICCIÓN DE COMPRAS A NIVELES PROMEDIOS

```python
#Llamamos a las librerías a usar
import pandas as pd
import numpy as np
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_error, r2_score
from sklearn.preprocessing import LabelEncoder
import warnings
warnings.filterwarnings("ignore")
import matplotlib.pyplot as plt
from sklearn.tree import plot_tree
```

```python
#Cargamos los datos
from google.colab import files
uploaded = files.upload()
```

```
Elegir archivos  Sin archivos seleccionados Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving Michela_y_Renata_con_Criticidad.xlsx to Michela_y_Renata_con_Criticidad.xlsx
```

```python
#Leemos los datos
df = pd.read_excel("Michela_y_Renata_con_Criticidad.xlsx", sheet_name="Sheet1")
```

```python
df
```

Mostrar salida oculta

```python
#Seleccionar variables relevantes
features = [
    'Criticidad',
    'Ventas',
    'Stock_Calculado',
    'Impacto_Monetario',
    'Inconsistencia_Stock',
    'categorias'
]
target = 'Compras'

X = df[features].copy()
y = df[target]
```

```python
#Codificación de variables categóricas

# Criticidad → ordinal (Alto=2, Medio=1, Bajo=0)
crit_map = {'Bajo': 0, 'Medio': 1, 'Alto': 2}
X['Criticidad'] = X['Criticidad'].map(crit_map)

# Categorías → target encoding (mejor que one-hot para árboles y pocos datos por categoría)
cat_means = df.groupby('categorias')['Compras'].mean()
X['categorias_encoded'] = X['categorias'].map(cat_means)

# Eliminar columna original
X = X.drop(columns=['categorias'])
```

```python
# División entrenamiento/prueba
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42)
```

```python
#Entrenar Random Forest
rf = RandomForestRegressor(
    n_estimators=200,
    max_depth=5,
    min_samples_split=10,
    min_samples_leaf=5,
    random_state=42,
    n_jobs=-1
)
rf.fit(X_train, y_train)
```

```
RandomForestRegressor(max_depth=5, min_samples_leaf=5, min_samples_split=10,
                      n_estimators=200, n_jobs=-1, random_state=42)
```

```python
#Predicción y métricas
y_pred = rf.predict(X_test)
mae = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print("✅ Modelo entrenado.")
print(f"📊 MAE (Error absoluto medio): {mae:.2f} unidades")
print(f"📈 R² (Coeficiente de determinación): {r2:.3f}")
```

```
✅ Modelo entrenado.
📊 MAE (Error absoluto medio): 1.18 unidades
📈 R² (Coeficiente de determinación): 0.943
```

```python
#Importancia de variables
importances = pd.Series(rf.feature_importances_, index=X.columns).sort_values(ascending=False)
print("\n🔍 Importancia de variables:")
print(importances)
```

```
🔍 Importancia de variables:
Ventas                0.872309
Stock_Calculado       0.102237
Criticidad            0.022842
Impacto_Monetario     0.001358
Inconsistencia_Stock  0.001100
categorias_encoded    0.000155
dtype: float64
```

```
#Visualizar el primer árbol (índice 0)
plt.figure(figsize=(30, 10))
plot_tree(
    rf.estimators_[0],
    feature_names=X.columns,
    filled=True,
    rounded=True,
    fontsize=10
)
plt.title("Árbol #0 del Random Forest (Profundidad máxima = 5)")
plt.savefig("random_forest_arbol_0.png", dpi=150, bbox_inches='tight')
plt.show()
```

Mostrar salida oculta

```
#Poner las predicciones en todo el archivo original
df['Predicciones'] = rf.predict(X)
```

```
df
```

Mostrar salida oculta

DECLARACIÓN Y AUTORIZACIÓN

Yo, **Chica Vega, María Renata**, con C.C: # **0707274130** autora del trabajo de titulación: **Aplicación del Machine Learning para la clasificación y predicción e importación de repuestos tipo crítico para automotores** previo a la obtención del título de **Licenciada en Negocios Internacionales** en la Universidad Católica de Santiago de Guayaquil.

1.- Declaro tener pleno conocimiento de la obligación que tienen las instituciones de educación superior, de conformidad con el Artículo 144 de la Ley Orgánica de Educación Superior, de entregar a la SENESCYT en formato digital una copia del referido trabajo de titulación para que sea integrado al Sistema Nacional de Información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.

2.- Autorizo a la SENESCYT a tener una copia del referido trabajo de titulación, con el propósito de generar un repositorio que democratice la información, respetando las políticas de propiedad intelectual vigentes.

Guayaquil, 13 de febrero de 2026

f. _____

Chica Vega, María Renata

C.C: 0707274130

DECLARACIÓN Y AUTORIZACIÓN

Yo, **Sucuzhanay Ceron, Michela Francesca** con C.C: **# 1723131551** autora del trabajo de titulación: **Aplicación del machine learning para la clasificación y predicción e importación de repuestos tipo crítico para automotores** previo a la obtención del título de **Licenciada en Negocios Internacionales** en la Universidad Católica de Santiago de Guayaquil.

1.- Declaro tener pleno conocimiento de la obligación que tienen las instituciones de educación superior, de conformidad con el Artículo 144 de la Ley Orgánica de Educación Superior, de entregar a la SENESCYT en formato digital una copia del referido trabajo de titulación para que sea integrado al Sistema Nacional de Información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.

2.- Autorizo a la SENESCYT a tener una copia del referido trabajo de titulación, con el propósito de generar un repositorio que democratice la información, respetando las políticas de propiedad intelectual vigentes.

Guayaquil, 13 de febrero de 2026

f. _____

Sucuzhanay Ceron, Michela Francesca

C.C: 1723131551

REPOSITORIO NACIONAL EN CIENCIA Y TECNOLOGÍA

FICHA DE REGISTRO DE TESIS/TRABAJO DE TITULACIÓN

TEMA Y SUBTEMA:	Aplicación del machine learning para la clasificación y predicción e importación de repuestos tipo crítico para automotores.		
AUTOR(ES)	Chica Vega, María Renata Sucuzhanay Ceron, Michela Francesca		
REVISOR(ES)/TUTOR(ES)	**Carrera Buri, Félix Miguel**		
INSTITUCIÓN:	Universidad Católica de Santiago de Guayaquil		
FACULTAD:	Facultad de Economía y Empresa		
CARRERA:	Negocios Internacionales		
TITULO OBTENIDO:	Licenciada en Negocios Internacionales		
FECHA DE PUBLICACIÓN:	13 de febrero de 2026	**No. DE PÁGINAS:**	**115 p.**
ÁREAS TEMÁTICAS:	Inteligencia Artificial, Data Science, Gestión de Inventarios, Balanza comercial, Importaciones, Industria automotriz, Comercio internacional.		
PALABRAS CLAVES/ KEYWORDS:	Machine Learning, gestión de inventarios, repuestos críticos, K-Means, Random Forest.		

RESUMEN/ABSTRACT:

La presente investigación tuvo como objetivo implementar un modelo de *Machine Learning* para la clasificación y predicción de importación de repuestos tipo crítico en el sector automotriz. La problemática se fundamenta en la dependencia de métodos tradicionales de gestión de inventarios en pequeñas y medianas empresas, lo que genera ineficiencias en la cadena de suministro, sobrecostos operativos y dificultades para anticipar la demanda. Metodológicamente, se aplicó un modelo de aprendizaje no supervisado K-Means para clasificar los repuestos según su nivel de criticidad (alto, medio y bajo), considerando variables como rotación, impacto monetario, stock calculado e inconsistencia de inventario. Posteriormente, se desarrolló un modelo predictivo Random Forest Regressor para estimar la variable compras, evaluando su desempeño mediante el Error Absoluto Medio (MAE) y el coeficiente de determinación (R^2). Los resultados evidenciaron una alta concentración de repuestos clasificados como críticos y un desempeño predictivo óptimo del modelo, con un MAE de 1.18 unidades y un R^2 de 0.943, lo que indica una elevada capacidad explicativa. Se concluye que la aplicación de técnicas de aprendizaje automático mejora significativamente la gestión de inventarios, optimiza la planificación de importaciones y fortalece la competitividad empresarial en el sector automotriz.

ADJUNTO PDF:	☒ SI	☐ NO
CONTACTO CON AUTOR/ES:	**Teléfono:** +593 982546528 +593 99 376 4200	**E-mail:** renata22chica@gamil.com michela1132b@gmail.com
CONTACTO CON LA INSTITUCIÓN (C00RDINADOR DEL PROCESO UTE)::	**Nombre: Freire Quintero César Enrique**	
	Teléfono: +593-990090702	
	E-mail: cesar.freire@cu.ucsg.edu.ec	

SECCIÓN PARA USO DE BIBLIOTECA

Nº. DE REGISTRO (en base a datos):	
Nº. DE CLASIFICACIÓN:	
DIRECCIÓN URL (tesis en la web):	