



INSTITUTO SUPERIOR UNIVERSITARIO TECNOLÓGICO DEL AZUAY

TECNOLOGÍA SUPERIOR EN BIG DATA

MINERÍA DE DATOS - APRENDIZAJE DE MÁQUINA

SISTEMA INTELIGENTE DE PREDICCIÓN DE INVENTARIOS CON BIG DATA PARA EL MERCADO DE CONSTRUCCIÓN

INTEGRANTES:

JIMMY SUMBA

FREDDY MONTALVAN

EDUARDO MENDIETA

DOCENTE:

ING. LADY SANGACHA

CUENCA, 14 DE FEBRERO DE 2025

ÍNDICE GENERAL

1. ANTECEDENTES.....	2
2. OBJETIVOS.....	3
2.1 Objetivo General.....	3
2.2 Objetivos específicos.....	3
3. INTRODUCCIÓN.....	4
4. INVESTIGACIONES SIMILARES.....	7
5. METODOLOGÍA.....	9
6. DESARROLLO.....	11
6.1 Fase 1: Comprensión del negocio.....	11
6.1.1 Definición de Requisitos.....	11
6.1.2 Diseño Conceptual.....	13
6.1.3 Diseño Lógico.....	15
6.1.4 Diseño Físico.....	17
6.1.5 Implementación.....	22
6.2 Fase 2: Comprensión de los datos.....	23
6.3 Fase 3: Preparación de los datos.....	24
6.3.1 Procesamiento y preprocesamiento de datos.....	24
6.3.2 Exploración de los datos.....	28
6.4 Fase 4: Modelado.....	32
6.5 Fase 5: Evaluación.....	38
6.6 Fase 6: Despliegue.....	41
7. RESULTADOS.....	43
7.1 Dataset para el Análisis de Ventas y Satisfacción del Cliente.....	43
7.2 Evaluación de los algoritmo para la Predicción de Inventarios.....	45
8. CONCLUSIONES.....	51
9. BIBLIOGRAFÍA.....	52

SISTEMA INTELIGENTE DE PREDICCIÓN DE INVENTARIOS CON BIG DATA PARA EL MERCADO DE CONSTRUCCIÓN

1. ANTECEDENTES

En la ciudad de Cuenca, la empresa MQ Distribuidora ha estado ofreciendo sus servicios de venta al por mayor y menor de diversos tipos de ladrillos, productos de construcción y ferretería desde el año 2000. Aunque sus operaciones se mantienen en registros de papel y su modelo de negocio ha funcionado satisfactoriamente, la empresa planea adaptarse a las exigencias del mundo tecnológico actual para aprovechar las ventajas que este ofrece. En este contexto, y gracias al auge de la inteligencia artificial, MQ Distribuidora ha decidido implementar algunas de las tecnologías disponibles en el mercado, con el objetivo de mejorar sus operaciones, aumentar la satisfacción del cliente y, por supuesto, obtener mayores ganancias.

Además, la empresa busca digitalizar su información, trasladándola de registros físicos a una base de datos que permita una mayor eficiencia en todas sus transacciones y optimice sus operaciones. Entre sus requerimientos se incluye la capacidad de predecir el número de productos que debe tener en stock en los diferentes meses, clasificados por tipo, considerando que algunos productos son elaborados de manera artesanal y pueden no venderse en ciertos períodos, lo que representa una pérdida. También desean medir el grado de satisfacción de los clientes para ofrecer un mejor servicio y evitar la pérdida de clientes potenciales en el futuro. Por todo lo anterior, nuestro equipo de desarrollo ha comenzado a trabajar en la construcción tanto de la base de datos como del sistema que permitirá prever el stock de productos y evaluar la satisfacción del cliente. Cabe destacar que toda la información ha sido proporcionada por los representantes legales de MQ Distribuidora.

2. OBJETIVOS

2.1 Objetivo General

- Predecir la demanda de productos de construcción para el análisis inteligente en las ventas y satisfacción de cliente que permita ampliar el negocio MQ distribuidora a través de herramientas tecnológicas.

2.2 Objetivos específicos

- Obtener una data set para el análisis inteligente en las ventas y satisfacción de cliente que permita ampliar el negocio MQ distribuidora.
- Modelar un algoritmo para la predicción de inventarios a través de modelos de machine learning con Python.
- Visualizar de forma interactiva el análisis inteligente de las ventas y satisfacción de cliente que permita ampliar el negocio MQ distribuidora.

3. INTRODUCCIÓN

En este informe se detalla el desarrollo de un esquema de base de datos, un sistema de predicción de stock mensual por producto y un análisis de satisfacción del cliente para la empresa MQ Distribuidora. A nivel internacional, el uso de machine learning en el sector de la construcción ha transformado la gestión de operaciones y la toma de decisiones. Estas tecnologías permiten analizar grandes volúmenes de datos, optimizando la gestión de inventarios, pronosticando la demanda y mejorando la eficiencia en la cadena de suministro, lo que reduce costos y mejora la satisfacción del cliente al asegurar la disponibilidad de productos. Además, el machine learning ayuda a identificar tendencias del mercado, permitiendo a las empresas adaptarse rápidamente a las condiciones cambiantes del sector (Gurtubay Regulez, A., 2021).

En Ecuador, el sector de la construcción está comenzando a adoptar el machine learning, aunque a un ritmo más lento. Las empresas locales pueden beneficiarse de estas tecnologías para optimizar sus procesos de venta y mejorar la gestión de proyectos. Por ejemplo, al analizar datos históricos de ventas, pueden predecir qué productos tendrán mayor demanda en diferentes épocas del año, ajustando su inventario y reduciendo el desperdicio. Además, el uso de machine learning en la evaluación de riesgos y la planificación de proyectos puede ayudar a minimizar costos y mejorar la seguridad en las obras, impulsando así el crecimiento del sector en la región (Li, H., & Zuo, J., 2019).

Diversas investigaciones han explorado la aplicación de metodologías de análisis predictivo en la gestión de inventarios. Flores Taquiri y Montalvo Celis (2024) implementaron un sistema predictivo en PYMES tecnológicas utilizando la metodología CRISP-DM, destacando la precisión del algoritmo de mejora gradual. Boada (2017) desarrolló una herramienta automatizada para prever la demanda en el sector cosmético, resaltando la importancia de la planificación estratégica. Moina Álvarez y Changoluisa Chillagana (2021) implementaron un prototipo de control de inventario en una fábrica de cueros, mejorando la eficiencia mediante inteligencia

artificial. Berenguel Velasquez (2019) presentó un sistema que utiliza aprendizaje automático para medir la satisfacción del cliente, logrando un modelo con alta exactitud. Finalmente, Morales Tabares (2016) propuso un modelo multivariado para predecir el stock de piezas de repuesto médicas, contribuyendo a una gestión más eficiente en el sector salud. Estas investigaciones evidencian la efectividad de los modelos de machine learning en la administración de inventarios y la satisfacción del cliente.

Se adoptó la metodología CRISP-DM para guiar el desarrollo del proyecto, estructurada en seis fases que facilitan la comprensión del negocio, la calidad de los datos y la alineación con los objetivos empresariales. En la primera fase, se trabajó con MQ Distribuidora para entender sus necesidades y establecer objetivos. La fase dos se centró en la construcción de un conjunto de datos en Excel y la exploración de variables utilizando Pandas en Python. En la fase tres, se realizó la limpieza y tratamiento de datos para prepararlos para el modelado. La fase cuatro consistió en seleccionar tres modelos de aprendizaje supervisado: Regresión Lineal, Árboles de Decisión y Random Forest, para comparar resultados. En la fase cinco, se evaluaron los modelos mediante técnicas como el Error Cuadrático Medio y el Coeficiente de Determinación. Finalmente, en la fase seis, aunque no se realizó un despliegue completo, se utilizaron herramientas como Power BI para presentar los resultados en un tablero de control, garantizando un enfoque sistemático y efectivo para el éxito del proyecto.

El desarrollo del Sistema Inteligente de Predicción de Inventarios se concretó en un dataset de 22 columnas, que incluye información sobre facturas, productos y clientes, con el objetivo de entrenar modelos de machine learning para predecir el stock y analizar la satisfacción del cliente. Se evaluaron tres modelos: Regresión Lineal, Árboles de Decisión y Random Forest, todos mostrando tendencias crecientes en la relación entre predicciones y valores reales, aunque con ajustes imperfectos. La distribución de errores reveló que la mayoría se concentraba alrededor de cero, indicando un buen ajuste general, aunque el modelo de Random Forest presentó más errores extremos. En Power BI, se determinó que la zona de

Cuenca Norte tenía la mayor satisfacción del cliente, con un predominio notable de hombres satisfechos (83.17%) frente a mujeres (16.83%).

4. INVESTIGACIONES SIMILARES

Para el desarrollo del presente proyecto, se ha encontrado que otras investigaciones utilizan metodologías diversas para el análisis predictivo, las cuales se describen a continuación.

Flores Taquiri y Montalvo Celis (2024) investigaron el uso de machine learning en la gestión de inventarios para PYMES dedicadas a la venta de productos tecnológicos. Su estudio, basado en la metodología CRISP-DM, demostró que la implementación de un sistema predictivo basado en algoritmos como la regresión lineal, árbol aleatorio y mejora gradual, permitió a la empresa mejorar la rotación de inventarios y reducir su duración. El algoritmo de mejora gradual se destacó por su precisión, evidenciada a través de un coeficiente de determinación (R^2) elevado y menores errores cuadráticos medios (MSE).

Por otro lado, Boada (2017) desarrolló un sistema de proyección de la demanda en una empresa de venta por catálogo en el sector cosmético. Este trabajo se centró en la creación de una herramienta automatizada que predice la demanda de productos, considerando variables tanto cuantitativas como cualitativas. La investigación subraya la importancia de la planificación estratégica y la automatización en la estimación de ventas, lo que resulta en una gestión de inventarios más eficiente.

Moina Álvarez y Changoluisa Chillagana (2021) diseñaron e implementaron un prototipo de control de inventario en una fábrica de cueros, utilizando inteligencia artificial. Su enfoque combinó métodos de observación y entrevistas para identificar debilidades en el control de inventarios existentes. La implementación de una aplicación web y un sistema de predicción de ventas resultó en mejoras significativas en la eficiencia del manejo de inventarios, destacando la efectividad del prototipo en la automatización de procesos.

Asimismo, Berenguel Velasquez (2019) presentó un sistema web que utiliza aprendizaje automático y análisis textual para detectar la satisfacción del cliente en MALL HOGAR S.A.C. La

investigación se basó en una muestra de 100 clientes y logró un modelo predictivo con una exactitud del 92.9%. Este estudio resalta la capacidad del aprendizaje automático para identificar patrones de satisfacción, lo que puede ser crucial para mejorar la atención al cliente.

Finalmente, Morales Tabares (2016) propuso un modelo multivariado para la predicción del stock de piezas de repuesto para equipos médicos. Su investigación abordó la incertidumbre en la demanda de estas piezas, utilizando técnicas de regresión lineal múltiple para mejorar la exactitud de los pronósticos. El modelo MPREDSTOCK, desarrollado como parte del Sistema de Gestión para Ingeniería Clínica y Electromedicina (SIGICEM), ofrece una guía para su implementación, contribuyendo así a la gestión eficiente del stock en el sector de la salud.

Estas investigaciones evidencian la adaptabilidad y eficacia de los modelos de aprendizaje automático en la administración de inventarios y la satisfacción del cliente, ofreciendo un contexto útil para el avance de este proyecto.

5. METODOLOGÍA

Se ha adoptado la metodología CRISP-DM (Cross-Industry Standard Process for Data Mining), la cual se estructura en seis etapas (Cortina, 2015). Utilizar CRISP-DM para nuestro proyecto resulta beneficioso debido a su clara y definida estructura, que guía a los equipos a través de las fases del proceso, desde la comprensión del negocio hasta el despliegue del modelo. Esta metodología enfatiza la importancia de la calidad de los datos y la alineación con los objetivos empresariales, permitiendo iteraciones flexibles entre fases. Además, facilita la evaluación y validación del modelo, asegura una documentación adecuada y mejora la comunicación entre los miembros del equipo, contribuyendo así al éxito del proyecto (Moine, Haedo, & Gordillo, 2011).

Fase 1: Comprensión del negocio

En esta fase, el equipo de desarrollo se reunió con la empresa MQ Distribuidora y su gerente, con el objetivo de entender el negocio y establecer las reglas necesarias para crear un modelo de base de datos que se ajuste a sus necesidades. También se identificó el objetivo del sistema de predicción (Sara, 2016).

Fase 2: Comprensión de los datos

Se construyó un conjunto de datos en Excel a partir de registros físicos. A través de esta herramienta, se realizó un análisis preliminar de las distintas variables. Posteriormente, los datos fueron montados y explorados con mayor profundidad utilizando Pandas en Python (Sara, 2016).

Fase 3: Preparación de los datos

Se llevó a cabo la limpieza y tratamiento del conjunto de datos, transformando los datos a su formato correcto en Pandas, preparando así la información para el modelado (Cortina, 2015).

Fase 4: Modelado

Se seleccionaron tres modelos de aprendizaje supervisado: Regresión Lineal, Árboles de Decisión para Regresiones y Random Forest. Esta selección se realizó con el objetivo de comparar los resultados obtenidos de los tres modelos (Cortina, 2015).

Fase 5: Evaluación

Se evaluaron los modelos aplicando técnicas como el Error Cuadrático Medio (MSE), el Coeficiente de Determinación, la Visualización de Predicciones y la Distribución de Errores. Estas técnicas permitieron determinar si el modelo lograba una predicción satisfactoria, evitando errores significativos (Cortina, 2015).

Fase 6: Despliegue

Aunque no se realizó un despliegue del sistema en sí, se utilizó la herramienta Power BI para presentar los resultados obtenidos en la predicción mediante un tablero de control (Cortina, 2015).

6. DESARROLLO

El desarrollo del Sistema Inteligente de Predicción de Inventarios, basado en Big Data, se llevó a cabo en seis etapas, las cuales se describen a continuación::

6.1 Fase 1: Comprensión del negocio

6.1.1 Definición de Requisitos

Para la creación de la base de datos, se llevó a cabo un levantamiento de información de forma presencial en la empresa. La persona encargada del negocio, quien posee un profundo conocimiento de las operaciones, nos guió a través de sus instalaciones y describió el flujo de sus procesos. A partir de estas observaciones, se redactaron las siguientes reglas de negocio:

La empresa MQ Distribuidora almacena productos en diferentes bodegas, donde cada producto se encuentra en una única bodega. Las bodegas pueden albergar distintos productos, y es fundamental conocer el estado de cada bodega, ya que se realizan mantenimientos periódicos. También es necesario saber la capacidad de almacenamiento de cada bodega.

En cuanto a los productos, se requiere almacenar la siguiente información:

- Nombre del producto
- Código asignado
- Precio de venta
- Descripción
- Categoría
- Stock actual
- Stock máximo
- Stock mínimo

Los clientes pueden adquirir diferentes productos, y se requiere almacenar su información personal, que incluye:

- CI/RUC
- Nombres
- Dirección
- Correo electrónico
- Teléfono de contacto

Las compras realizadas por los clientes pueden requerir despacho desde diferentes bodegas, lo cual implica el transporte de los productos hacia un destino específico. Cada bodega cuenta con un único empleado responsable de realizar los despachos, y es necesario registrar qué empleado transportó la mercancía comprada por el cliente.

La empresa dispone de varios vehículos, que son conducidos por diferentes empleados según la cantidad de producto adquirida, lo que permite que cualquier empleado pueda conducir cualquier vehículo. Para cada vehículo, es fundamental conocer:

- Matrícula
- Modelo
- Capacidad de carga

En relación a los empleados, se requiere almacenar información sobre:

- Cargo desempeñado
- Horario
- Salario

Además de producir diferentes tipos de ladrillos, la empresa también adquiere productos de terceros. Es esencial conocer la identidad del proveedor, ya que los productos se adquieren de diversas fuentes, y cada proveedor genera una factura que debe ser cancelada. Por lo tanto, es importante almacenar la información de las adquisiciones realizadas a los proveedores, que incluye:

- Nombre de la empresa
- Responsable
- Teléfono
- Dirección
- Correo electrónico

6.1.2 Diseño Conceptual

Basado en las reglas de negocio anteriormente descritas, se elabora un esquema Entidad-Relación (ER). Según Bermúdez León, M. J. (2020), el modelo entidad-relación es un enfoque conceptual para bases de datos relacionales que permite representar de manera gráfica las abstracciones y el conocimiento en un sistema de información. Utiliza un diagrama ER para ilustrar un conjunto de objetos llamados entidades y sus interacciones, reflejando de forma natural las relaciones del mundo real.

Importancia

La importancia de un diagrama ER radica en su capacidad para simplificar la complejidad de los datos (Gaona, A. L. (2012). Al proporcionar una vista clara y concisa de las entidades y sus relaciones, ayuda a:

- Identificar redundancias.
- Asegurar que se capturen todos los requisitos necesarios.
- Actuar como un plano para la construcción de la base de datos, guiando el proceso de diseño y evitando errores en etapas posteriores (Bermúdez León, M. J. 2020).

Componentes del Diagrama ER

Según Bermúdez León, M. J. (2020), un diagrama entidad-relación se compone de varias partes clave:

Entidades: Representadas como rectángulos, son los objetos o conceptos que se almacenan en la base de datos.

Atributos: Describen las características de las entidades y se muestran como óvalos conectados a sus respectivas entidades.

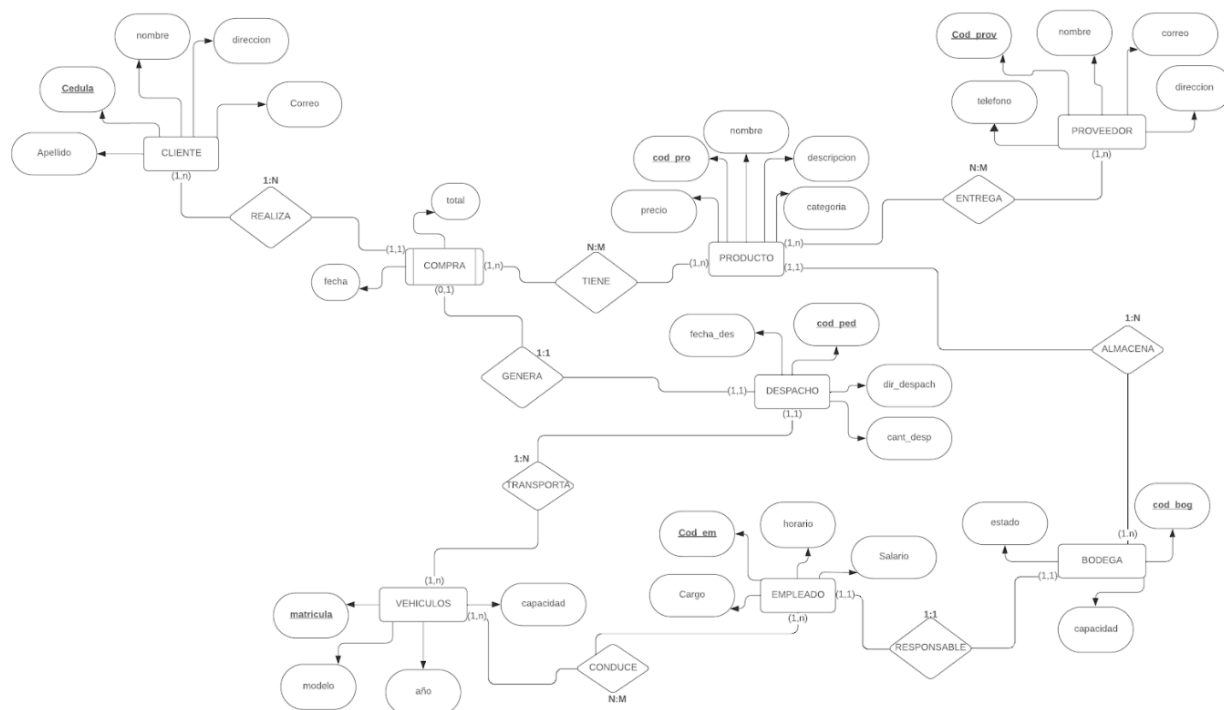
Relaciones: Indican cómo se asocian las entidades entre sí y se representan mediante rombos.

Cardinalidad: Define el número de instancias de una entidad que pueden estar asociadas con instancias de otra, lo que es crucial para entender la naturaleza de las interacciones en el sistema.

Las entidades base identificadas en las reglas del funcionamiento del negocio son:

- Cliente
- Compra
- Producto
- Proveedor
- Despacho
- Bodega
- Empleado
- Vehículo

Las relaciones y atributos, después del análisis, se pueden observar en la imagen a continuación:



6.1.3 Diseño Lógico

Una vez obtenido el diagrama entidad-relación, es necesario transformarlo en un esquema más formal denominado Modelo Relacional. Este modelo es una forma de estructurar y organizar datos en bases de datos relacionales, derivado del diagrama entidad-relación. Permite representar datos de manera tabular, donde cada entidad se traduce en una tabla y las relaciones entre entidades se reflejan a través de claves foráneas. Su objetivo es facilitar la gestión, recuperación y manipulación de datos de manera eficiente, manteniendo la integridad y coherencia de la información (Elmasri et al., 2007).

Importancia del Modelo Relacional

La generación de un modelo relacional es fundamental porque:

- Proporciona una base sólida para el diseño de bases de datos.
- Asegura que los datos estén organizados de forma lógica y accesible, facilitando su uso en aplicaciones y sistemas de información.
- Ayuda a identificar redundancias y establecer reglas de normalización, contribuyendo a una mejor calidad de los datos (Elmasri et al., 2007).

Componentes del Modelo Relacional

El modelo relacional se compone de:

- Tablas: Cada tabla representa una entidad y contiene filas (registros) y columnas (atributos).
- Claves Primarias: Identifican de manera única cada registro en una tabla.
- Claves Foráneas: Establecen las relaciones entre diferentes tablas.
- Restricciones y Reglas: Garantizan la integridad de los datos, como la unicidad y la obligatoriedad de ciertos campos, asegurando que la base de datos funcione correctamente (Elmasri et al., 2007).

Proceso de Normalización

Una vez construido el modelo relacional, este se normaliza mediante técnicas de normalización hasta la tercera forma normal (3NF) utilizando datos de prueba y una herramienta visual como Excel. Según Adrián, T. E. (2016), la normalización de bases de datos es un proceso estructural que organiza los datos para reducir la redundancia y mejorar la integridad de la información. Este proceso implica:

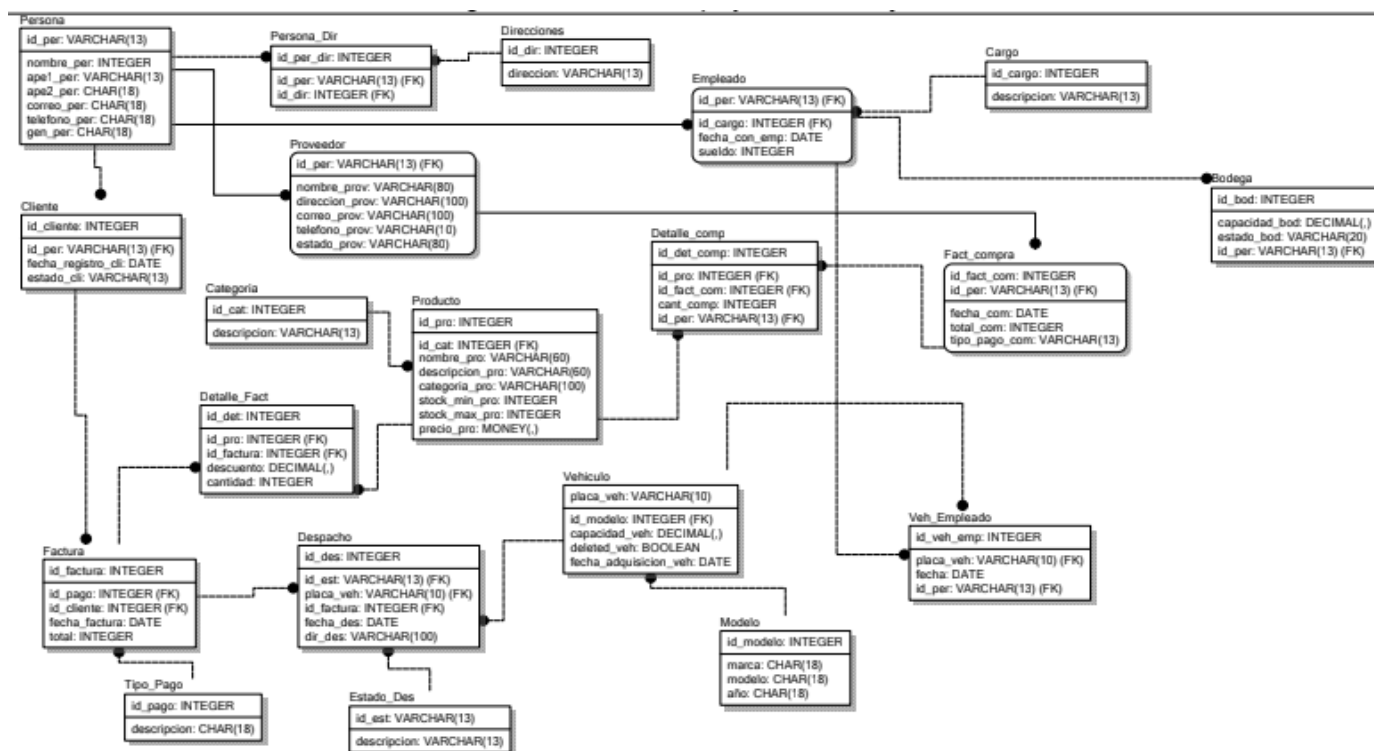
- Dividir grandes tablas en tablas más pequeñas.
- Definir relaciones entre ellas.

Formas Normales

- Primera Forma Normal (1NF): Todos los atributos de una tabla deben contener solo valores atómicos (sin conjuntos de valores). Cada registro debe ser único, requiriendo una clave primaria que identifique de manera única cada fila (Adrián, T. E., 2016).
- Segunda Forma Normal (2NF): Se logra cuando una tabla está en 1NF y todos los atributos no clave son completamente dependientes de la clave primaria. No debe haber dependencias parciales; si un atributo depende solo de una parte de una clave compuesta, debe moverse a otra tabla (Adrián, T. E., 2016).
- Tercera Forma Normal (3NF): Se alcanza cuando una tabla está en 2NF y todos los atributos no clave son independientes entre sí. No debe haber dependencias transitivas, y se deben crear nuevas tablas para aquellos atributos que dependen de otros (Adrián, T. E., 2016).

Resultado de la Normalización

Una vez concluido el proceso de normalización, se obtiene el Modelo Relacional Normalizado, que se observa en la imagen:



6.1.4 Diseño Físico

Una vez obtenido el modelo relacional normalizado, se procede a crear el esquema de la base de datos utilizando SQL. Este proceso inicia con la creación de las tablas, junto con sus respectivas relaciones y restricciones. Según Escofet (2002), el lenguaje SQL (Structured Query Language) es un estándar fundamental para gestionar y manipular bases de datos relacionales. Permite realizar diversas operaciones, como la creación, modificación y consulta de datos almacenados en tablas. SQL es esencial para interactuar con sistemas de gestión de bases de datos (SGBD), ya que proporciona una sintaxis clara y estructurada para ejecutar comandos

como SELECT, INSERT, UPDATE y DELETE, facilitando así la gestión eficiente de grandes volúmenes de información.

La importancia de SQL en la modelación de bases de datos radica en su capacidad para implementar el diseño conceptual de una base de datos en un entorno práctico. A través de SQL, los diseñadores pueden crear las estructuras necesarias, definir relaciones entre tablas y establecer restricciones de integridad. Esto no solo permite una organización adecuada de los datos, sino que también asegura que las consultas y transacciones se realicen de manera coherente y segura, lo cual es esencial para el funcionamiento de aplicaciones y sistemas que dependen de la información almacenada (Escofet, 2002).

El esquema SQL para la creación de tablas basado en el modelo relacional normalizado es el siguiente:

```
CREATE TABLE Persona (
    id_per VARCHAR(13) PRIMARY KEY,
    nombre_per VARCHAR(18),
    apel_per VARCHAR(18),
    ape2_per VARCHAR(18),
    correo_per VARCHAR(50),
    telefono_per VARCHAR(10),
    gen_per VARCHAR(18)
);

CREATE TABLE Direccion (
    id_direccion SERIAL PRIMARY KEY,
    direccion VARCHAR(200)
);

CREATE TABLE Cargo (
    id_cargo SERIAL PRIMARY KEY,
    descripcion_cargo VARCHAR(200)
);

CREATE TABLE Tipo_Pago (
    id_tp SERIAL PRIMARY KEY,
    descripcion_tp VARCHAR(200)
);
```

```

CREATE TABLE Estado_Des (
    id_ed SERIAL PRIMARY KEY,
    descripcion_ed VARCHAR(200)
);

CREATE TABLE Modelo (
    id_modelo SERIAL PRIMARY KEY,
    marca VARCHAR(20),
    modelo VARCHAR(20),
    anio INTEGER
);

CREATE TABLE Categoria (
    id_cat SERIAL PRIMARY KEY,
    descripcion_cat VARCHAR(200)
);

CREATE TABLE Persona_Dir (
    id_per_dir SERIAL PRIMARY KEY,
    id_per VARCHAR(13),
    id_dir INTEGER,
    FOREIGN KEY (id_per) REFERENCES Persona (id_per) ON DELETE
    CASCADE,
    FOREIGN KEY (id_dir) REFERENCES Direccion (id_direccion) ON
    DELETE CASCADE
);

CREATE TABLE Cliente (
    id_cliente SERIAL PRIMARY KEY,
    id_per VARCHAR(13),
    fecha_registro_cli DATE,
    estado_cli VARCHAR(13),
    FOREIGN KEY (id_per) REFERENCES Persona (id_per) ON DELETE
    CASCADE
);

CREATE TABLE Proveedor(
    id_proveedor VARCHAR(13) PRIMARY KEY,
    nombre_prov VARCHAR(80),
    direccion_prov VARCHAR(100),
    correo_prov VARCHAR(100),
    telefono_prov VARCHAR(10),
    estado_prov VARCHAR(80),
    FOREIGN KEY (id_proveedor) REFERENCES Persona (id_per)
);

CREATE TABLE Empleado (
    id_emp VARCHAR(13) PRIMARY KEY,
    id_cargo INTEGER,
    fecha_con_emp DATE,
    sueldo NUMERIC(10, 2),

```

```

        FOREIGN KEY (id_emp) REFERENCES Persona (id_per) ON DELETE
        CASCADE,
        FOREIGN KEY (id_cargo) REFERENCES Cargo (id_cargo) ON DELETE
        CASCADE
    );

```

```

CREATE TABLE Bodega (
    id_bodega SERIAL PRIMARY KEY,
    capacidad_bodega NUMERIC(10, 2),
    estado_bodega VARCHAR(20),
    id_emp VARCHAR (13),
    FOREIGN KEY (id_emp) REFERENCES Empleado (id_emp) ON DELETE
    CASCADE
);

```

```

CREATE TABLE Fac_Compra (
    id_fac_compra SERIAL PRIMARY KEY,
    id_proveedor VARCHAR(13),
    id_tipo_pago INTEGER,
    fecha_compra DATE,
    total_compra NUMERIC(10, 2),
    FOREIGN KEY (id_proveedor) REFERENCES Proveedor (id_proveedor) ON
    DELETE CASCADE,
    FOREIGN KEY (id_tipo_pago) REFERENCES Tipo_Pago (id_tp) ON DELETE
    CASCADE
);

```

```

CREATE TABLE Producto (
    id_pro SERIAL PRIMARY KEY,
    id_categoria INTEGER,
    nombre_pro VARCHAR(100),
    descripcion_pro VARCHAR(200),
    stock_min_pro INTEGER,
    stock_max_pro INTEGER,
    stock_pro INTEGER,
    precio_pro NUMERIC(10, 2),
    FOREIGN KEY (id_categoria) REFERENCES Categoria (id_cat) ON
    DELETE CASCADE
);

```

```

CREATE TABLE Detalle_Compra (
    id_det_compra SERIAL PRIMARY KEY,
    id_producto INTEGER,
    id_fac_compra INTEGER,
    cantidad_compra INTEGER,
    precio_unitario_compra NUMERIC(10, 2),
    FOREIGN KEY (id_producto) REFERENCES Producto (id_pro) ON DELETE
    CASCADE,
    FOREIGN KEY (id_fac_compra) REFERENCES Fac_Compra (id_fac_compra)
    ON DELETE CASCADE
);

```

```

CREATE TABLE Factura (
    id_factura SERIAL PRIMARY KEY,
    id_cliente INTEGER,
    id_tipo_pago INTEGER,
    fecha_factura DATE,
    total NUMERIC(10,2),
    FOREIGN KEY (id_cliente) REFERENCES Cliente (id_cliente) ON
DELETE CASCADE,
    FOREIGN KEY (id_tipo_pago) REFERENCES Tipo_Pago (id_tp) ON DELETE
CASCADE
);

```

```

CREATE TABLE Detalle_Fact (
    id_det SERIAL PRIMARY KEY,
    id_producto INTEGER,
    id_factura INTEGER,
    iva_det NUMERIC(3, 2) CHECK (iva_det >= 0 AND iva_det <= 1),
    descuento_det NUMERIC(3, 2) CHECK (descuento_det >= 0 AND
descuento_det <= 1),
    cantidad_det INTEGER,
    precio_unitario_det NUMERIC(10, 2),
    FOREIGN KEY (id_producto) REFERENCES Producto (id_pro) ON DELETE
CASCADE,
    FOREIGN KEY (id_factura) REFERENCES Factura (id_factura) ON
DELETE CASCADE
);

```

```

CREATE TABLE Vehiculo (
    placa_vehi VARCHAR(10) PRIMARY KEY,
    id_modelo INTEGER,
    capacidad_vehi NUMERIC(10, 2),
    deleted_vehi BOOLEAN,
    fecha_adquisicion_vehi DATE,
    FOREIGN KEY (id_modelo) REFERENCES Modelo (id_modelo) ON DELETE
CASCADE
);

```

```

CREATE TABLE Despacho (
    id_despacho INTEGER PRIMARY KEY,
    id_estado INTEGER,
    placa_vehi VARCHAR(10),
    id_factura INTEGER,
    fecha_despacho DATE,
    direccion_despacho VARCHAR(100),
    FOREIGN KEY (id_estado) REFERENCES Estado_Des (id_ed) ON DELETE
CASCADE,
    FOREIGN KEY (placa_vehi) REFERENCES Vehiculo (placa_vehi) ON
DELETE CASCADE,
    FOREIGN KEY (id_factura) REFERENCES Factura (id_factura) ON
DELETE CASCADE

```

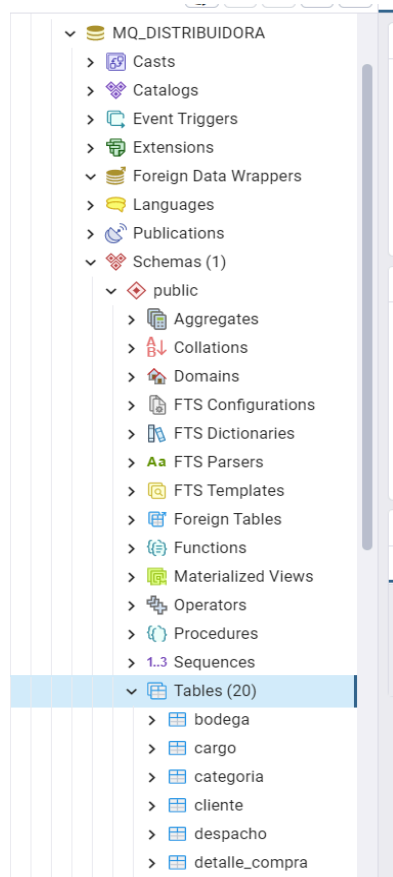
```
);
```

```
CREATE TABLE Vehi_Empleado (
    id_vehi_empleado SERIAL PRIMARY KEY,
    id_empleado VARCHAR(13),
    placa_vehi VARCHAR(10),
    fecha_uso_vehi DATE,
    FOREIGN KEY (id_empleado) REFERENCES Empleado (id_emp) ON DELETE
    CASCADE,
    FOREIGN KEY (placa_vehi) REFERENCES Vehiculo (placa_vehi) ON
    DELETE CASCADE
);
```

6.1.5 Implementación

La creación de la base de datos se llevó a cabo en el sistema de gestión de bases de datos PostgreSQL, donde se insertaron varios datos de prueba y se realizaron consultas de verificación. Este sistema fue seleccionado debido a que, según Ginesta y Mora (2012), PostgreSQL es un sistema relacional de código abierto que ofrece características avanzadas, tales como transacciones ACID, soporte para tipos de datos complejos y la posibilidad de extensibilidad mediante funciones personalizadas y procedimientos almacenados.

Adicionalmente, PostgreSQL es reconocido por su capacidad para manejar grandes volúmenes de datos y su rendimiento en entornos de alta concurrencia, lo que lo convierte en una solución ideal para aplicaciones empresariales críticas y proyectos de desarrollo que requieren escalabilidad y seguridad. Su robustez y versatilidad lo posicionan como una opción preferida para diversas necesidades en la gestión de datos, resultando de gran utilidad para el desarrollo de este proyecto.



6.2 Fase 2: Comprensión de los datos

Para el desarrollo de la base de datos, se contó con todas las descripciones sobre el modo de operación y las reglas del negocio proporcionadas por la empresa, lo que permitió concluir con dicho entregable. Sin embargo, para diseñar el modelo de predicción del stock mensual de productos y el análisis de la satisfacción del cliente, no se tuvo la misma suerte. Gran parte de la información proporcionada correspondía a facturas de ventas de diferentes años y al número de productos mantenidos en stock a lo largo de los meses.

Proceso de Construcción del Dataset

Se procedió a construir un dataset utilizando el formato de archivo Excel para migrar los datos desde los factureros y registros en papel. Este proceso estuvo acompañado de un diccionario de datos para cada columna en el mismo archivo. Se eligió Excel por su facilidad de uso, interactividad y las funcionalidades que ayudaron en la preparación de los datos.

Cada columna cuenta con su respectiva descripción, lo que facilita la comprensión de los datos y su posterior análisis.

Este enfoque metódico y estructurado fue fundamental para asegurar la calidad y utilidad de los datos en el desarrollo del proyecto.

6.3 Fase 3: Preparación de los datos

En esta fase del proyecto, utilizamos Google Colab debido a sus ventajas en colaboración y facilidad de uso.

6.3.1 Procesamiento y preprocesamiento de datos

Dado que los datos fueron ingresados manualmente por los diferentes integrantes del equipo en un dataset de Excel, este presentaba numerosos errores, como campos nulos, errores de escritura y registros con múltiples campos en blanco. Para abordar estas inconsistencias, es necesario pasar por una etapa de procesamiento y preprocesamiento de datos, asegurando que la información esté limpia y lista para el modelamiento.

Según Rodríguez Rodríguez (2010), el preprocesamiento de datos en minería de datos es una etapa crucial que implica la preparación y limpieza de los datos antes de su análisis. Este proceso incluye actividades como la eliminación de duplicados, el manejo de valores faltantes, la normalización y transformación de datos, así como la reducción de dimensionalidad. El

objetivo del preprocesamiento es mejorar la calidad de los datos, garantizando que sean consistentes y relevantes para el análisis posterior. Un adecuado preprocesamiento incrementa la precisión y efectividad de los modelos de minería de datos, resultando en mejores insights y decisiones basadas en datos.

Herramientas Utilizadas

Para la preparación de los datos se utilizó Python y sus bibliotecas esenciales para el tratamiento y visualización de datos: Numpy, Pandas, Matplotlib y Seaborn. En primera instancia, se cargan estas bibliotecas para su posterior uso, como se puede observar en la imagen:

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

Los datos se transforman en un dataset de Pandas, y en esta línea de código también se convierten datos categóricos que Pandas interpreta como numéricos, como se muestra en la imagen:

```
data = pd.read_excel('dataset.xlsx', dtype={'rucCICli': str, 'telefono': str, 'climaPrecipitaciones': float, 'mes': int})
```

Técnicas de Preprocesamiento

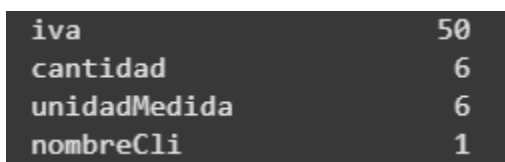
Rodríguez Rodríguez (2010) menciona varias técnicas para realizar el preprocesamiento de datos, entre las cuales se incluyen:

- Limpieza de Datos: Proceso de identificación y corrección de errores, inconsistencias y datos faltantes. Esto incluye la eliminación de duplicados, corrección de errores tipográficos y gestión de valores ausentes.

- **Transformación de Datos:** Modificación de datos para hacerlos más adecuados para el análisis, que puede incluir técnicas como normalización y codificación de variables categóricas en formatos numéricos. También puede incluir la creación de nuevas variables a partir de las existentes para resaltar patrones y relaciones.
- **Reducción de Dimensionalidad:** Disminución de la cantidad de datos a analizar sin perder información relevante, lograda mediante técnicas como el muestreo y la eliminación de características redundantes o irrelevantes. Esto mejora la eficiencia de los algoritmos de minería de datos y facilita la visualización y el análisis.

Inspección de Datos

Una vez cargados los datos, se realiza una inspección rápida utilizando comandos como head, shape, isnull y sum, que proporcionan información básica pero crucial para el análisis y determinan qué técnica es apropiada para el tratamiento. Es importante destacar que parte del tratamiento mediante limpieza de datos se realizó en Excel debido a su facilidad para filtrar datos y corregir errores, como los errores en la estructura gramatical de los nombres de los productos. En la información sobre datos nulos, se observa que existen cuatro columnas con valores nulos, como se ilustra en la imagen:



iva	50
cantidad	6
unidadMedida	6
nombreCli	1

Filtrado de Valores Nulos

Para filtrar estos valores nulos, se aplica la técnica de imputación de data cleaning, reemplazando valores desconocidos numéricos, como la cantidad de productos vendidos, por cero, y otros, como el IVA, por 0.12, que ha sido el estándar durante muchos años. Para datos categóricos, como la unidad de medida y el nombre del cliente, se reemplaza por la palabra “desconocido”. En el caso de los errores en el código del producto, se sustituye toda la columna

por un nuevo código basado en el nombre del producto, su categoría y una numeración. Para clasificar la unidad de medida, se crea un diccionario que indica a cada unidad de medida el tipo correspondiente, y luego se mapea toda la columna de tipo, como se observa en la imagen:

```
tipos = {  
    'PIEZA': 'VOLUMEN',  
    'METRO': 'LONGITUD',  
    'CAJA(20PZ)': 'VOLUMEN',  
    'CARRERA': 'TIEMPO',  
}  
  
df.tipo = df.unidadMedida.map(tipos)  
df.tipo = df.tipo.fillna('DESCONOCIDO')
```

Corrección de Tipos de Datos

Otro problema identificado fue que la columna de cantidad se estaba tratando como objeto. Para corregir esto, se recurre a la técnica de Data Transformation utilizando Pandas, forzando la conversión de la columna a entero y reemplazando los valores nulos por cero, como se muestra en la imagen 8.

```
df.cantidad = pd.to_numeric(df.cantidad, errors='coerce')  
df.cantidad = df.cantidad.fillna(0)
```

Creación de Nuevas Variables

Por último, se crea una nueva columna llamada “año” a partir de la columna de “fechaEmisión”, que será útil más adelante para ejecutar los modelos.

6.3.2 Exploración de los datos

Para obtener una visión más general sobre la información del dataset, es fundamental realizar un análisis exploratorio. Pandas facilita este trabajo a través de un conjunto variado de funciones diseñadas para este propósito. A continuación, se presentan algunas de las funciones más útiles:

- **df.head():** Muestra las primeras 5 filas del DataFrame, útil para obtener una vista previa rápida de los datos.
- **df.tail():** Muestra las últimas 5 filas del DataFrame.
- **df.info():** Proporciona un resumen del DataFrame, incluyendo el número de entradas, tipos de datos y valores no nulos.
- **df.describe():** Genera estadísticas descriptivas para las columnas numéricas, como media, desviación estándar, mínimo y máximo.
- **df.columns:** Devuelve una lista de los nombres de las columnas en el DataFrame.
- **df.shape:** Muestra las dimensiones del DataFrame (número de filas y columnas).
- **df.isnull().sum():** Muestra el número de valores nulos en cada columna del DataFrame.
- **df['columna'].unique():** Devuelve los valores únicos de una columna específica.
- **df['columna'].value_counts():** Agrupa el DataFrame por una columna y calcula la media de las otras columnas.
- **df.corr():** Calcula la matriz de correlación entre las columnas numéricas del DataFrame.
- **df['columna'].hist():** Crea un histograma de los valores en una columna específica.
- **df.plot(kind='scatter', x='col1', y='col2'):** Crea un gráfico de dispersión entre dos columnas.
- **df['columna'].describe(include='all'):** Genera estadísticas descriptivas para todas las columnas, incluidas las categóricas.
- **df.sample(n=10):** Muestra una muestra aleatoria de 10 filas del DataFrame.

Gráficas

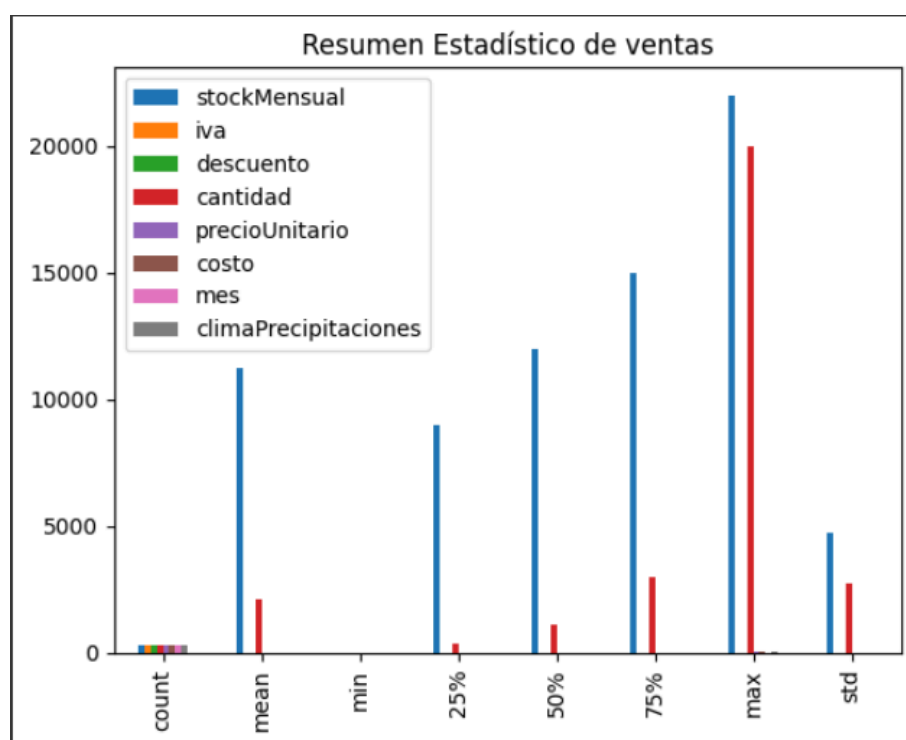
A continuación, se muestran algunas gráficas obtenidas a partir de este análisis exploratorio utilizando bibliotecas como Matplotlib y Seaborn:

Histogramas: Para visualizar la distribución de una variable numérica.

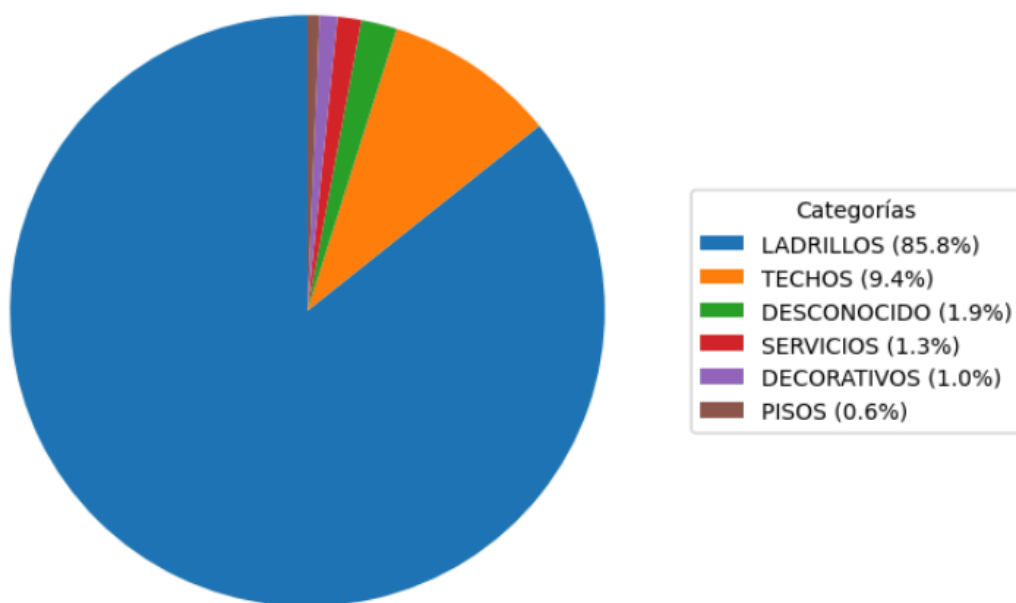
Gráficos de dispersión: Para explorar la relación entre dos variables.

Gráficas de barras: Para comparar frecuencias de categorías.

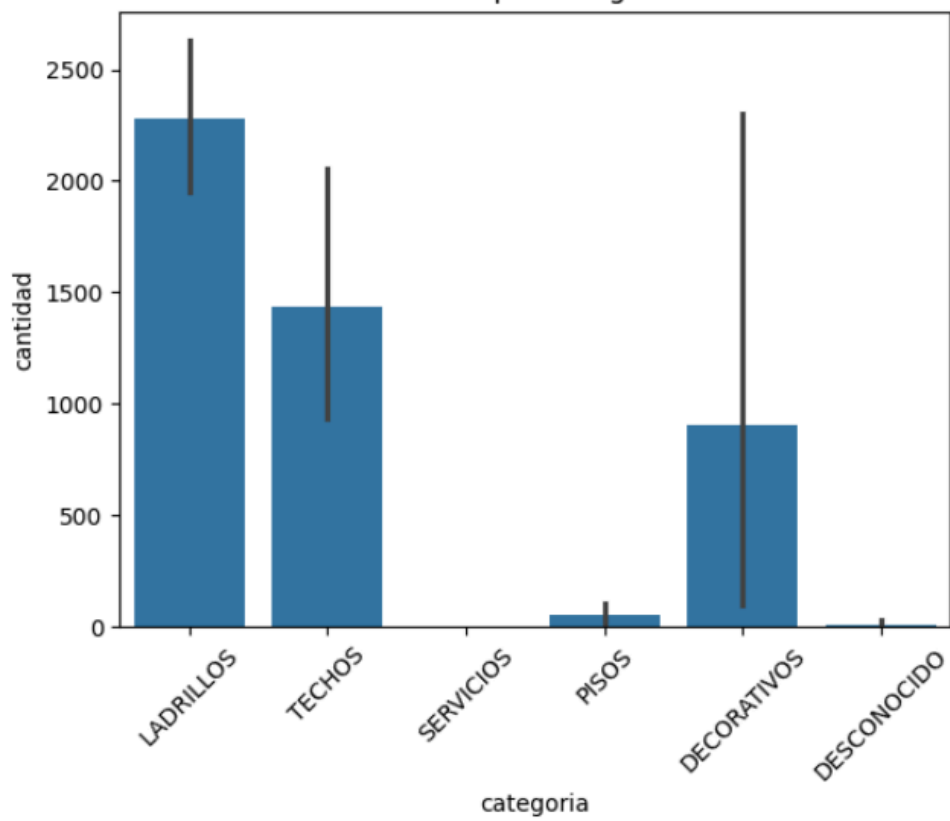
Estas visualizaciones son esenciales para comprender mejor los patrones y tendencias dentro de los datos, así como para identificar posibles anomalías o áreas que requieren atención adicional.

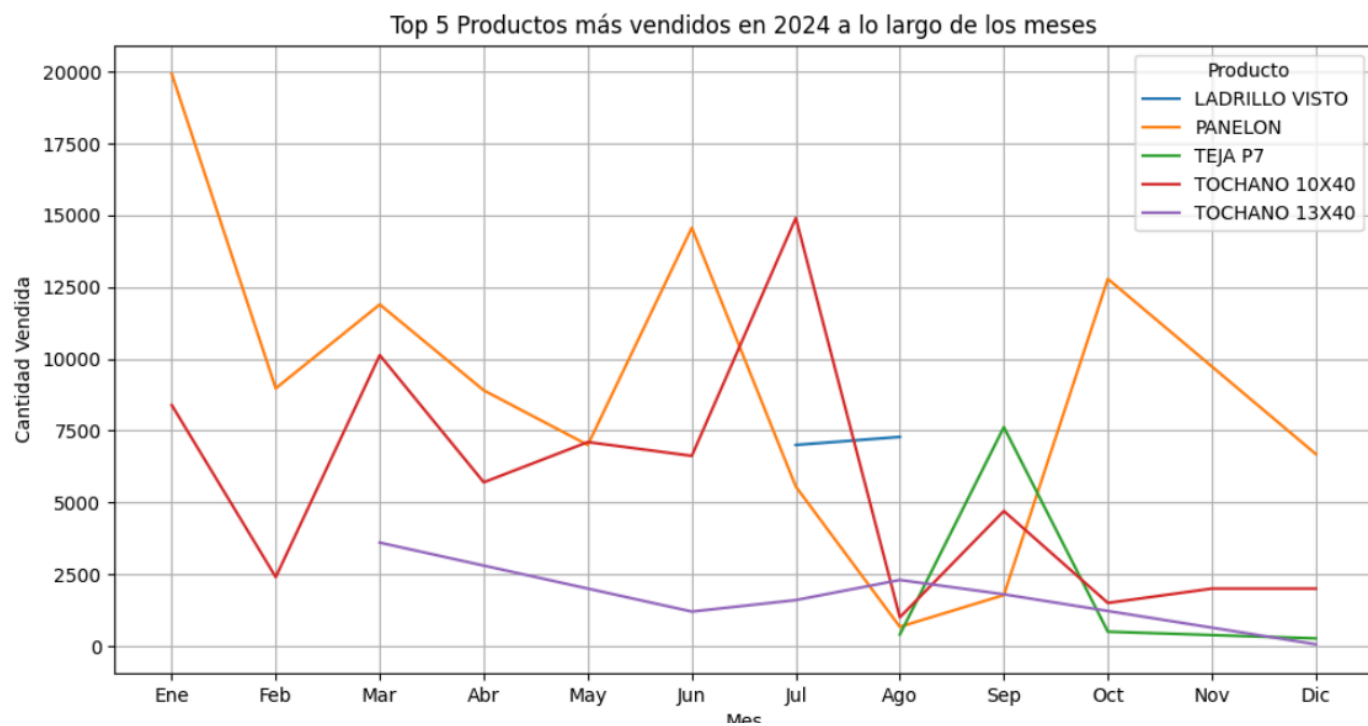
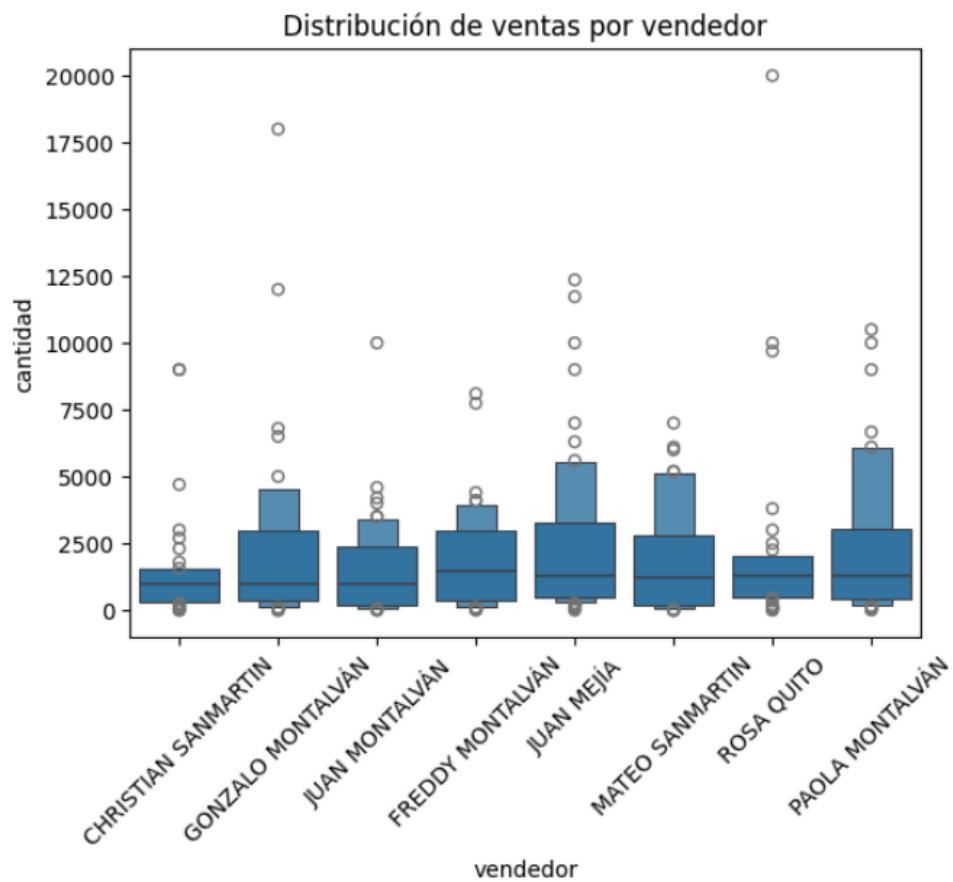


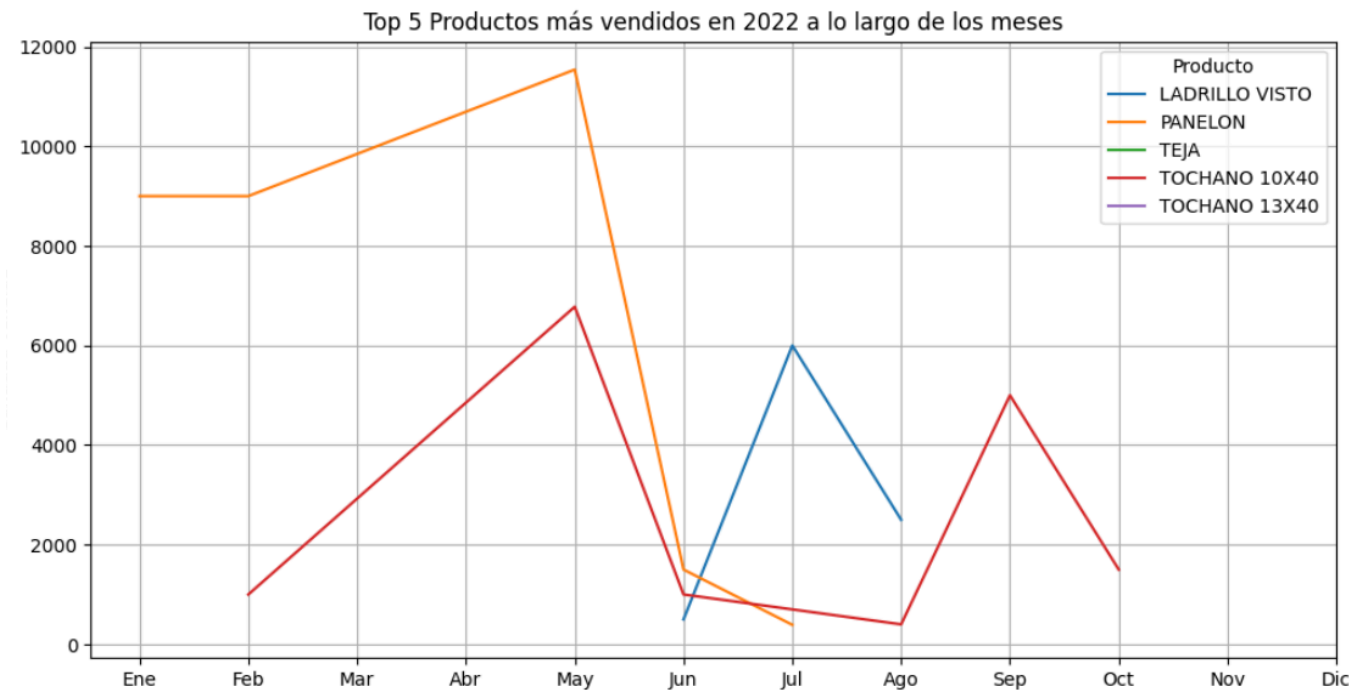
Proporción de ventas por categoría



Ventas por categoría







6.4 Fase 4: Modelado

Para lograr el objetivo de predecir el stock mensual por producto, hemos seleccionado un conjunto específico de variables independientes que ofrecen un mejor ajuste al modelo, lo que resulta en predicciones más precisas. Es importante mencionar que se probaron otras variables, como la cantidad de unidades vendidas, el precio de venta, los descuentos y la fecha de venta. Sin embargo, estas variables no lograron superar el 20% de efectividad en las predicciones. Tras un análisis más profundo, concluimos que no eran necesarias para la predicción.

Variables Independientes

- **nombreProducto:** Esencial para la predicción del stock por producto.
- **cantidad:** Indica el número total de unidades vendidas y es un fuerte predictor de la demanda. Generalmente, una mayor cantidad vendida en meses anteriores se correlaciona con un mayor stock necesario en el futuro.

- **mes:** Necesaria para predecir el stock mensual. La estacionalidad es un factor clave en la demanda de productos, ya que algunos pueden venderse más en ciertos meses del año.
- **año:** Ayuda a identificar tendencias a largo plazo, enriqueciendo el conjunto de datos y mejorando la precisión de las predicciones de inventario.

Variable Dependiente

- **stockMensual:** Cantidad total de productos disponibles en inventario al final de cada mes, que es el objetivo de nuestra predicción.

Modelos Seleccionados

Regresión Lineal: Permite predecir una variable continua a partir de variables independientes. Es útil para identificar relaciones lineales que afectan la demanda, como el precio y la estacionalidad. Su simplicidad y facilidad de interpretación ayudan a los gestores a comprender rápidamente cómo influyen los cambios en las variables en la cantidad de productos necesarios (Dagnino, J., 2014).

Árboles de Decisión: Dividen los datos en subconjuntos basándose en características específicas, facilitando la interpretación de decisiones. Son ideales para analizar factores como promociones y tipos de productos, y su capacidad para visualizar decisiones ayuda a identificar patrones complejos en la demanda, optimizando así la disponibilidad de productos (Arana, C., 2021).

Random Forest: Combina múltiples árboles de decisión para mejorar la precisión de las predicciones. Su capacidad para manejar grandes conjuntos de datos y múltiples variables reduce el riesgo de sobreajuste. Al combinar predicciones, ofrece estimaciones más robustas, esenciales para mantener niveles óptimos de stock y satisfacer la demanda del mercado (Espinosa-Zúñiga, J. J., 2020).

Preparación del Entorno

En un nuevo notebook de Google Colab, comenzamos cargando las librerías necesarias para trabajar con datos y realizar análisis. Utilizamos:

- **Numpy:** Para arreglos de datos.
- **Pandas:** Para manipulación y análisis de datos.
- **Matplotlib y Seaborn:** Para crear gráficos y visualizaciones, ayudándonos a representar los datos de forma visual y a identificar patrones o tendencias.
- **Además, empleamos las librerías de Scikit-learn** para los modelos de regresión lineal, árboles de decisión y Random Forest, que se utilizan para hacer predicciones basadas en datos. Esto incluye funciones para dividir los datos en conjuntos de entrenamiento y prueba, así como métricas para evaluar el rendimiento de los modelos. También implementamos técnicas como la codificación de variables categóricas y la creación de flujos de trabajo eficientes mediante pipelines, simplificando el proceso de análisis y modelado.

Proceso de Modelado

A continuación, cargamos los datos procesados en Pandas, asegurándonos de indicar los formatos de datos categóricos para evitar su conversión a números.

Se desarrolla una función que abstrae el proceso de modelado, permitiendo la reutilización de código. Este proceso general de modelado en machine learning se describe a través de la función `obtener_prediccion`. Comienza con la preparación de los datos, dividiéndolos en conjuntos de entrenamiento y prueba mediante `train_test_split`, que separa los datos en dos partes: una para entrenar el modelo y otra para evaluar su rendimiento. El parámetro `test_size` determina qué proporción de los datos se utilizará para la prueba, asegurando que el modelo se entrene con una cantidad suficiente de datos y se evalúe adecuadamente.

Una vez que los datos están divididos, el siguiente paso es entrenar el modelo utilizando el método `fit`, donde el modelo se ajusta a los datos de entrenamiento (`X_train` y `y_train`). Durante este proceso, el modelo aprende las relaciones y patrones presentes en los datos, lo que le permitirá hacer predicciones sobre nuevos datos en el futuro. Es fundamental que el modelo sea adecuado para el tipo de datos y el problema que se está abordando.

Después de entrenar el modelo, se realizan predicciones sobre el conjunto de prueba utilizando el método `predict`, que toma los datos de prueba (`X_test`) y genera las predicciones correspondientes (`y_pred`). Este paso es crucial para evaluar cómo de bien el modelo ha aprendido a partir de los datos de entrenamiento y si puede generalizar a datos no vistos.

Una parte importante del proceso es la comparación entre las predicciones y los valores reales. Se crea un nuevo DataFrame que incluye tanto los valores reales (`y_test`) como las predicciones generadas (`y_pred`), lo que permite visualizar y analizar el rendimiento del modelo, facilitando la identificación de posibles errores o áreas de mejora. Esta comparación es esencial para entender la efectividad del modelo y si cumple con los objetivos establecidos.

Finalmente, la función devuelve dos objetos: el DataFrame con las predicciones y los valores reales, que se utilizarán más tarde en un dashboard con las proyecciones, y el conjunto original de datos para la evaluación mediante métricas y gráficas.

```
def obtener_prediccion(model, X, y, test_size):  
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=test_size, random_state=42)  
  
    model.fit(X_train, y_train)  
  
    y_pred = model.predict(X_test)  
  
    res = X_test.copy()  
  
    res['y_test'] = y_test  
    res['y_pred'] = y_pred  
  
    return res, y
```

Preparación de Datos para el Análisis

Se preparan las variables `x_stock` y `y_stock`, donde `x_stock` contiene las características de entrada (nombre del producto, mes y año) y `y_stock` representa la variable objetivo, que es el stock mensual. Se crea un objeto `ColumnTransformer` llamado `preprocessor_stock`, que se utiliza para preprocesar los datos. Este transformador aplica diferentes técnicas a distintas columnas: utiliza `OneHotEncoder` para convertir la columna categórica `nombreProducto` en variables indicadoras, permitiendo que el modelo maneje adecuadamente las categorías, y deja pasar sin cambios las columnas numéricas `mes` y `anio`, que no requieren transformación. Esto asegura que los datos estén en un formato adecuado para el modelado posterior.

```
x_stock = df[['nombreProducto', 'mes', 'anio']]
y_stock = df['stockMensual']

preprocessor_stock = ColumnTransformer(
    transformers=[
        ('cat', OneHotEncoder(handle_unknown='ignore'), ['nombreProducto']),
        ('num', 'passthrough', ['mes', 'anio'])
    ])

```

Ejecución de Modelos

A continuación, se aplican los conjuntos de datos de las variables dependientes e independientes, ya preprocesadas y transformadas, a los diferentes modelos:

Para los tres modelos, se utiliza un Pipeline de Scikit-learn, que permite encadenar múltiples etapas de procesamiento de datos y modelos en un solo objeto, simplificando el código y mejorando su organización. Facilita la aplicación de transformaciones como escalado y selección de características, asegurando que se realicen correctamente para evitar fugas de datos entre los conjuntos de entrenamiento y prueba. Además, permite una validación cruzada más sencilla y la reutilización de los mismos pasos en diferentes experimentos, optimizando el flujo de trabajo en el desarrollo de modelos de machine learning.

Regresión Lineal: Se llama a la función `obtener_prediccion` para hacer predicciones sobre los datos de entrada `x_stock` y `y_stock`, utilizando un tamaño de prueba del 20% (0.2). Se muestra el encabezado de las predicciones obtenidas accediendo a la primera parte del resultado con `pred_stock_rlineal[0].head()`.

```
model = Pipeline(steps=[
    ('preprocessor', preprocessor_stock),
    ('regressor', LinearRegression())
])

pred_stock_rlineal = obtener_prediccion(model, x_stock, y_stock, 0.2)

pred_stock_rlineal[0].head()
```

Árboles de Decisión: Se llama a la función `obtener_prediccion` para generar predicciones sobre los datos de entrada `x_stock` y `y_stock`, utilizando un tamaño de prueba del 15% (0.15). Se accede al encabezado de las predicciones obtenidas mostrando las primeras filas del resultado con `pred_stock_ardecision[0].head()`.

```
model = Pipeline(steps=[
    ('preprocessor', preprocessor_stock),
    ('regressor', DecisionTreeRegressor(random_state=42))
])

pred_stock_ardecision = obtener_prediccion(model, x_stock, y_stock, 0.15)

pred_stock_ardecision[0].head()
```

Random Forest: Se llama a la función `obtener_prediccion` para realizar predicciones sobre los datos de entrada `x_stock` y `y_stock`, utilizando un tamaño de prueba del 15% (0.15). Se muestra el encabezado de las predicciones generadas accediendo a las primeras filas del resultado con `pred_stock_rforest[0].head()`.

```

model = Pipeline(steps=[
    ('preprocessor', preprocessor_stock),
    ('regressor', RandomForestRegressor(n_estimators=100, random_state=42))
])

pred_stock_rforest = obtener_prediccion(model, x_stock, y_stock, 0.15)

pred_stock_rforest[0].head()

```

Una vez obtenidos los resultados de las predicciones, se procederá a su evaluación en la siguiente sección.

6.5 Fase 5: Evaluación

Cajahuanca, J. E. V., Raymundo, Á. F. N., Franco, A. C. L., y Flores, J. D. J. (2022) destacan que evaluar un modelo de machine learning mediante métricas es esencial para comprender su rendimiento y efectividad en la tarea específica abordada. Las métricas permiten cuantificar la precisión de las predicciones y facilitan la comparación entre diferentes modelos o configuraciones. Sin una evaluación adecuada, resulta complicado determinar si un modelo está funcionando correctamente o si necesita ajustes. Además, las métricas ayudan a identificar problemas como el sobreajuste o el subajuste, lo que permite a los desarrolladores tomar decisiones informadas sobre cómo mejorar el modelo.

Una de las métricas más utilizadas en la evaluación de modelos de regresión es el Error Absoluto Medio (MAE, por sus siglas en inglés). El MAE mide la media de los errores absolutos entre las predicciones y los valores reales, proporcionando una medida clara de la precisión del modelo en términos de unidades originales. Esta métrica es fácil de interpretar y no penaliza desproporcionadamente los errores grandes, lo que la hace útil en situaciones donde se desea una evaluación robusta sin influencias extremas de valores atípicos.

Otra métrica importante es el Error Cuadrático Medio (MSE), que calcula la media de los cuadrados de los errores entre las predicciones y los valores reales. A diferencia del MAE, el MSE penaliza más fuertemente los errores grandes debido al uso del cuadrado, lo que lo

convierte en una métrica sensible a valores atípicos. Esta característica puede ser ventajosa en situaciones donde se desea minimizar los grandes errores, aunque también puede llevar a una evaluación sesgada si hay muchos valores atípicos presentes en los datos.

El coeficiente de determinación (R^2 Score) es otra métrica crucial que indica la proporción de la varianza en la variable dependiente que es predecible a partir de las variables independientes. Un R^2 Score de 1 indica que el modelo explica completamente la variabilidad de la respuesta, mientras que un valor de 0 sugiere que el modelo no proporciona ninguna información útil. Esta métrica es especialmente valiosa para comparar modelos, ya que permite entender qué tan bien se ajusta un modelo a los datos en comparación con un modelo simple que solo predice la media. En conjunto, el MAE, el MSE y el R^2 Score ofrecen una visión integral del rendimiento del modelo y son herramientas clave en el proceso de optimización y evaluación.

Con los resultados de las predicciones, estamos en posición de evaluar el desempeño de los modelos. Para ello, se han preparado las siguientes funciones:

Función `evaluar_modelo`: Calcula y muestra el rendimiento de un modelo de predicción utilizando tres métricas de error: el Error Absoluto Medio (MAE), el Error Cuadrático Medio (MSE) y el R^2 Score. Esta función toma como entrada una lista que contiene un objeto con los valores reales (`y_test`) y las predicciones (`y_pred`). A través de las funciones de la biblioteca `sklearn.metrics`, se calculan las métricas y se imprimen en un formato claro y preciso, mostrando cuatro decimales para mayor exactitud. Esto permite evaluar la efectividad del modelo en la predicción de datos.

```
def evaluar_modelo(model_pred_res):  
    mse = mean_squared_error(model_pred_res[0].y_test, model_pred_res[0].y_pred)  
    mae = mean_absolute_error(model_pred_res[0].y_test, model_pred_res[0].y_pred)  
    r2 = r2_score(model_pred_res[0].y_test, model_pred_res[0].y_pred)  
  
    print(f'Error Absoluto Medio (MAE): {mae}')  
    print(f'Error Cuadrático Medio (MSE): {mse}')  
    print(f'R2 Score: {r2}')
```


Función `graficar_pred_vs_reales`: Visualiza la comparación entre las predicciones de un modelo y los valores reales. Toma como entrada `model_pred_res`, que contiene las predicciones y los valores reales, y `lbl_modelo`, que es una etiqueta para identificar el modelo. Dentro de la función, se extraen las predicciones y los valores reales, creando una gráfica de dispersión donde se representan las predicciones en el eje vertical y los valores reales en el eje horizontal. Además, se traza una línea de referencia diagonal en rojo que indica dónde deberían coincidir las predicciones con los valores reales. Finalmente, se etiquetan los ejes, se añade un título que incluye el nombre del modelo, se muestra una leyenda y se presenta la gráfica en pantalla.

```
def graficar_pred_vs_reales(model_pred_res, lbl_modelo):
    pred = model_pred_res[0]
    y = model_pred_res[1]

    plt.figure(figsize=(10, 6))
    plt.scatter(pred.y_test, pred.y_pred, color='blue', label='Predicciones')
    plt.plot([y.min(), y.max()], [y.min(), y.max()], color='red', linestyle='--', label='Línea de referencia')
    plt.xlabel('Valores Reales')
    plt.ylabel('Valores Predichos')
    plt.title(f'Predicciones vs. Valores Reales en el modelo {lbl_modelo}')
    plt.legend()
    plt.show()
```

Función `graficar_distrib_err`: Visualiza la distribución de los errores de predicción de un modelo. Toma como entrada `model_pred_res`, que contiene las predicciones, y calcula los errores restando las predicciones (`pred.y_pred`) de los valores reales (`pred.y_test`). Luego, se crea una gráfica de histograma utilizando Seaborn para mostrar la frecuencia de los errores, añadiendo una curva de densidad (KDE) para representar la distribución de manera más suave. Se etiquetan los ejes con "Error" y "Frecuencia", y se añade un título que indica que se está mostrando la distribución de errores. Finalmente, se presenta la gráfica en pantalla.

```
def graficar_distrib_err(model_pred_res):
    pred = model_pred_res[0]

    errors = pred.y_test - pred.y_pred

    plt.figure(figsize=(10, 6))
    sns.histplot(errors, kde=True)
    plt.xlabel('Error')
    plt.ylabel('Frecuencia')
    plt.title('Distribución de Errores')
    plt.show()
```

Utilizando estas tres funciones, que representan una forma de evaluar cada uno de los modelos, se procede a pasar los resultados obtenidos en la predicción de cada uno de los modelos, como se observa en las imágenes a continuación.

----- Evaluación de modelo de Regresión Lineal:

Error Absoluto Medio (MAE): 2010.0835424309366
Error Cuadrático Medio (MSE): 6578162.202493794
R² Score: 0.8100977319877652

----- Evaluación de modelo de Árboles de Decisión:

Error Absoluto Medio (MAE): 687.8723404255319
Error Cuadrático Medio (MSE): 4264904.255319149
R² Score: 0.8686618225049902

----- Evaluación de modelo Random Forest:

Error Absoluto Medio (MAE): 913.791489361702
Error Cuadrático Medio (MSE): 3675998.3672340424
R² Score: 0.8867972415969225

6.6 Fase 6: Despliegue

A partir de la base de datos y de los resultados obtenidos en el análisis predictivo mediante los modelos Lineal, Random Forest y Árbol de Decisión, se desarrolló un dashboard utilizando la herramienta Microsoft Power BI. En esta plataforma se cargaron los archivos generados a partir de la base de datos y las proyecciones en formato .xlsx, lo que permitió conectar las columnas correspondientes de los archivos y relacionarlas con los resultados obtenidos en nuestro análisis.

Tras este proceso, se seleccionaron gráficos adecuados para presentar de forma clara los resultados. La primera pestaña del dashboard se centra en los datos estadísticos resultantes del

análisis y las proyecciones. Estos gráficos son fundamentales para comprender e interpretar el comportamiento de los datos. Por ello, se optó por incluir las siguientes visualizaciones:

- Compras realizadas por género.
- Stock promedio de los productos vendidos.
- Relación entre la cantidad de productos adquiridos, el precio y las precipitaciones.
- Análisis de las cantidades máximas y mínimas por producto en las distintas zonas de ventas.

En la pestaña dedicada a las proyecciones, se incluyó información derivada de los modelos Random Forest y Árbol de Decisión, con el propósito de identificar el stock necesario para el mes seleccionado. Este análisis se aplicó por producto, facilitando una mejor toma de decisiones respecto a las compras requeridas.

Además, esta sección permite conocer el nivel de satisfacción por género en relación con el producto seleccionado. También se presenta un segundo gráfico que refleja la satisfacción por zona respecto a la calidad de los productos adquiridos, proporcionando valiosa información para el análisis estratégico.

7. RESULTADOS

Los resultados obtenidos se enumeran a continuación. Los archivos del proyecto se pueden encontrar en el siguiente enlace: [GitHub - PF-S2](#).

7.1 Dataset para el Análisis de Ventas y Satisfacción del Cliente

A partir de la recolección de datos realizada en la empresa MQ Distribuidora y su posterior migración a Excel, se obtuvo un dataset compuesto por 22 columnas. El objetivo principal fue seleccionar las columnas adecuadas para el entrenamiento del modelo de predicción de stock y el análisis de satisfacción del cliente.

Estructura del Dataset

El dataset contiene las siguientes 22 columnas:

- **codFactura** (Object, Cualitativa Nominal): Identificador único de cada factura.
- **fechaEmision** (Object, Cualitativa Nominal): Fecha de emisión de la factura.
- **stockMensual** (Numérica, Cuantitativa Discreta): Cantidad total de productos disponibles al final de cada mes.
- **rucCICli** (Object, Cualitativa Nominal): Registro único del contribuyente del cliente.
- **nombreCli** (Object, Cualitativa Nominal): Nombre del cliente.
- **telefono** (Object, Cualitativa Nominal): Número de contacto del cliente.
- **direccion** (Object, Cualitativa Nominal): Ubicación del cliente.
- **zona** (Object, Cualitativa Nominal): División geográfica del cliente.
- **correo** (Object, Cualitativa Nominal): Correo electrónico del cliente.
- **formaPago** (Object, Cualitativa Nominal): Método de pago utilizado.
- **iva** (Numérica, Cuantitativa Continua): Impuesto al Valor Agregado.
- **descuento** (Numérica, Cuantitativa Continua): Reducción en el precio total.
- **codProducto** (Object, Cualitativa Nominal): Identificador único del producto.
- **nombreProducto** (Object, Cualitativa Nominal): Nombre del producto.

- categoria (Object, Cualitativa Nominal): Clasificación del producto.
- unidadMedida (Object, Cualitativa Nominal): Forma de medir y vender el producto.
- tipo (Object, Cualitativa Nominal): Clasificación de la unidad de medida.
- materiaPrima (Object, Cualitativa Nominal): Material fundamental en la fabricación.
- cantidad (Numérica, Cuantitativa Continua): Total de unidades vendidas.
- precioUnitario (Numérica, Cuantitativa Continua): Costo de una unidad.
- costo (Numérica, Cuantitativa Continua): Gasto total para producir o adquirir el producto.
- vendedor (Object, Cualitativa Nominal): Responsable de la venta.
- climaPrecipitaciones (Numérica, Cuantitativa Continua): Información sobre el clima en el mes de la venta.
- anio (Numérica, Cuantitativa Discreta): Año de la venta.

7.2 Evaluación de los algoritmo para la Predicción de Inventarios

Mediante Métricas

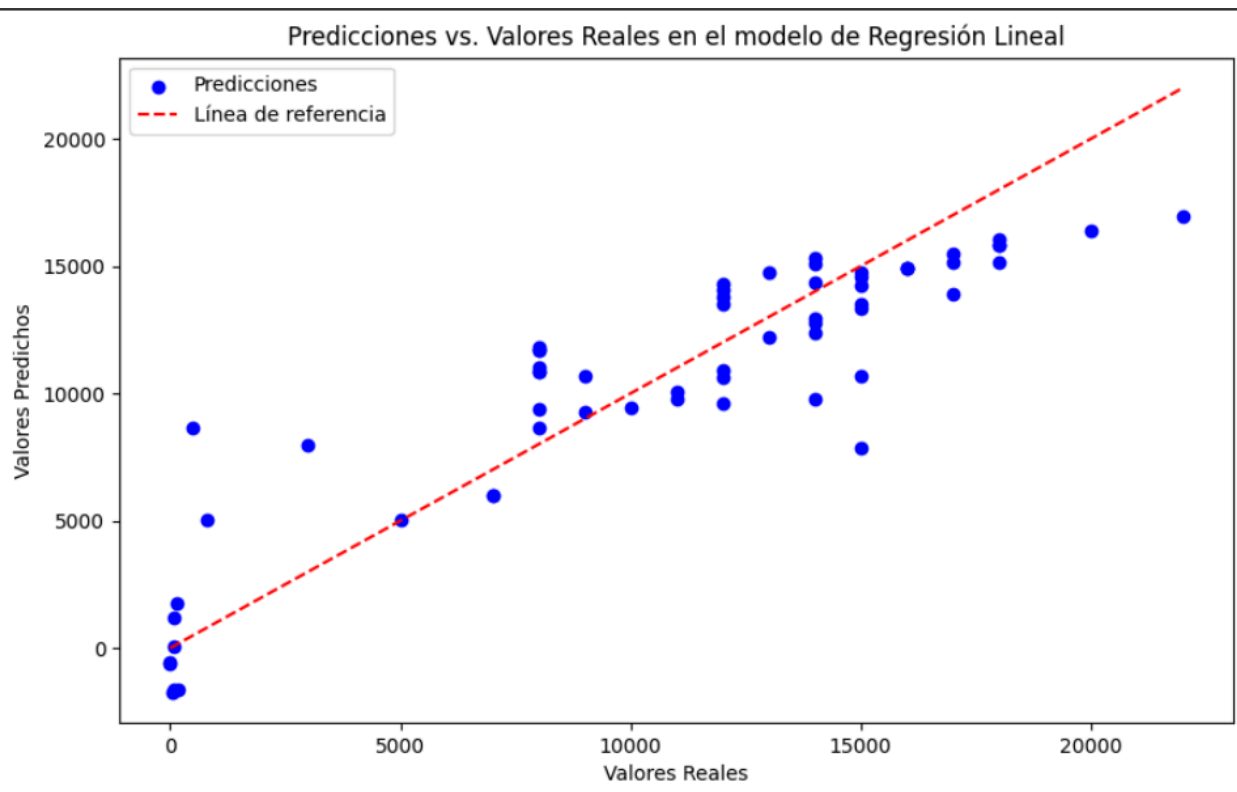
Del entrenamiento de los modelos de machine learning se obtuvieron los siguientes resultados en las evaluaciones, utilizando diversas métricas:

- Regresión Lineal:
 - Error Absoluto Medio (MAE): Es una métrica que mide la diferencia promedio entre los valores predichos y los valores reales. Un MAE más bajo indica un mejor ajuste del modelo.
 - Error Cuadrático Medio (MSE): Mide el promedio de los errores al cuadrado. Es más sensible a valores atípicos que el MAE. Un MSE más bajo también indica un mejor ajuste.
 - R2 Score: Es una métrica que indica qué tan bien el modelo se ajusta a los datos. Varía entre 0 y 1, donde 1 significa un ajuste perfecto. Un valor más alto de R2 es deseable.
- Árboles de Decisión:
 - MAE: 687.87, más bajo que el de Regresión Lineal, lo que indica un mejor ajuste.
 - MSE: 4264904.26, también más bajo que el de Regresión Lineal.
 - R2 Score: 0.87, más alto que el de Regresión Lineal, lo que significa que el modelo de Árboles de Decisión se ajusta mejor a los datos.
- Random Forest:
 - MAE: 913.79, más alto que el de Árboles de Decisión, pero aún así mejor que Regresión Lineal.
 - MSE: 3675998.37, el más bajo de los tres modelos.
 - R2 Score: 0.89, el más alto de los tres, lo que indica que el modelo de Random Forest es el que mejor se ajusta a los datos.

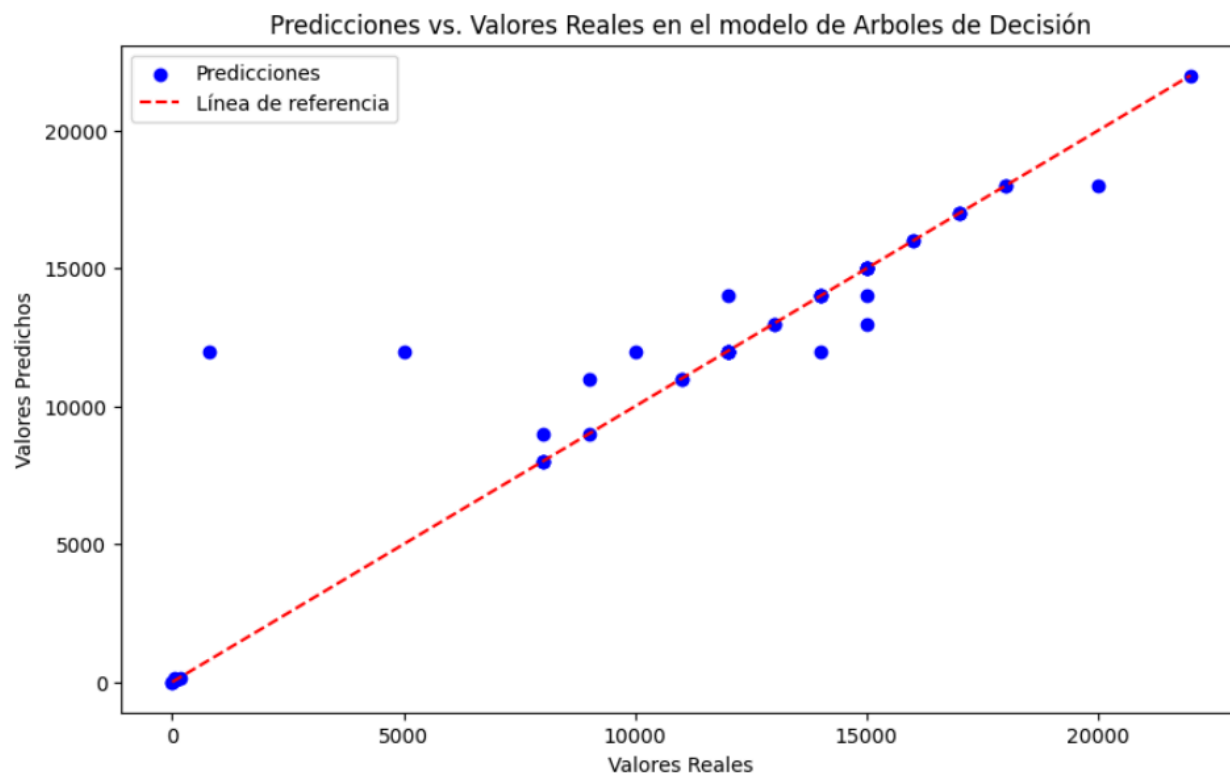
Siendo Random Forest el modelo que mejor se ajusta a los datos, con el menor error y el mayor coeficiente de determinación (R2). Lo cual sugiere que el modelo de Random Forest sería el más adecuado para realizar predicciones en este caso.

Evaluación de Predicciones vs. Valores Reales

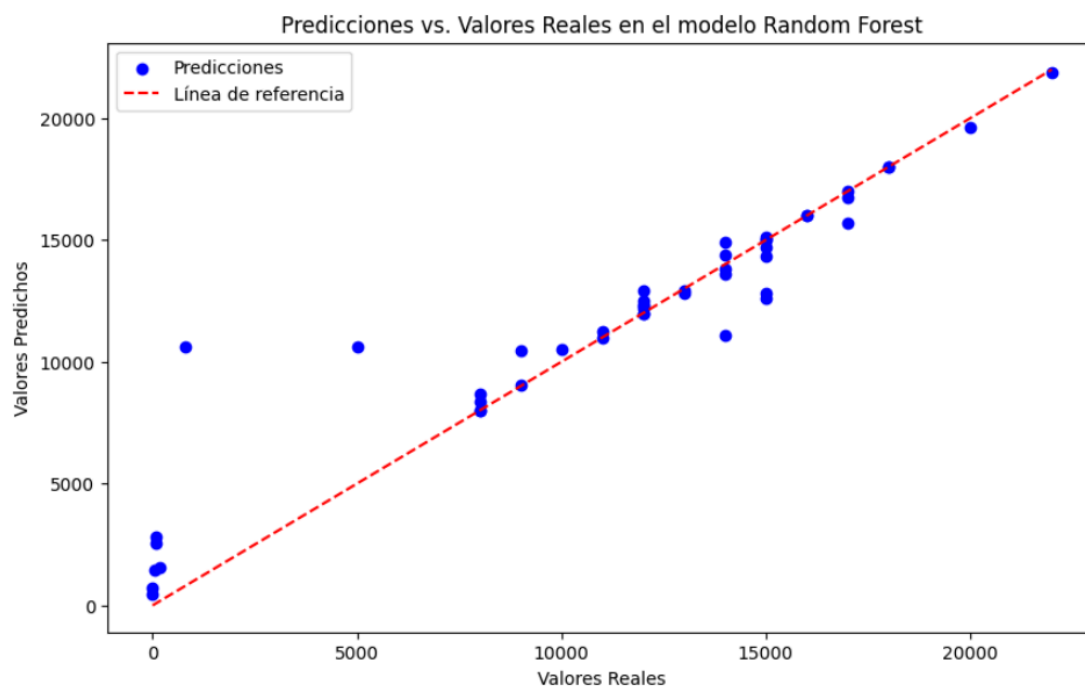
Modelo de Regresión Lineal: Los puntos azules representan los pares de valores predichos y reales, mostrando una tendencia creciente. Esto indica que el modelo logra capturar la relación entre las variables; sin embargo, algunos puntos se desvían de la línea de referencia, sugiriendo que el ajuste no es perfecto.



Modelo de Árboles de Decisión: Se observa una tendencia creciente, lo que indica que el modelo capta adecuadamente la relación entre las variables. No obstante, algunos puntos se desvían de la línea de referencia (línea roja punteada), lo que sugiere un ajuste imperfecto.

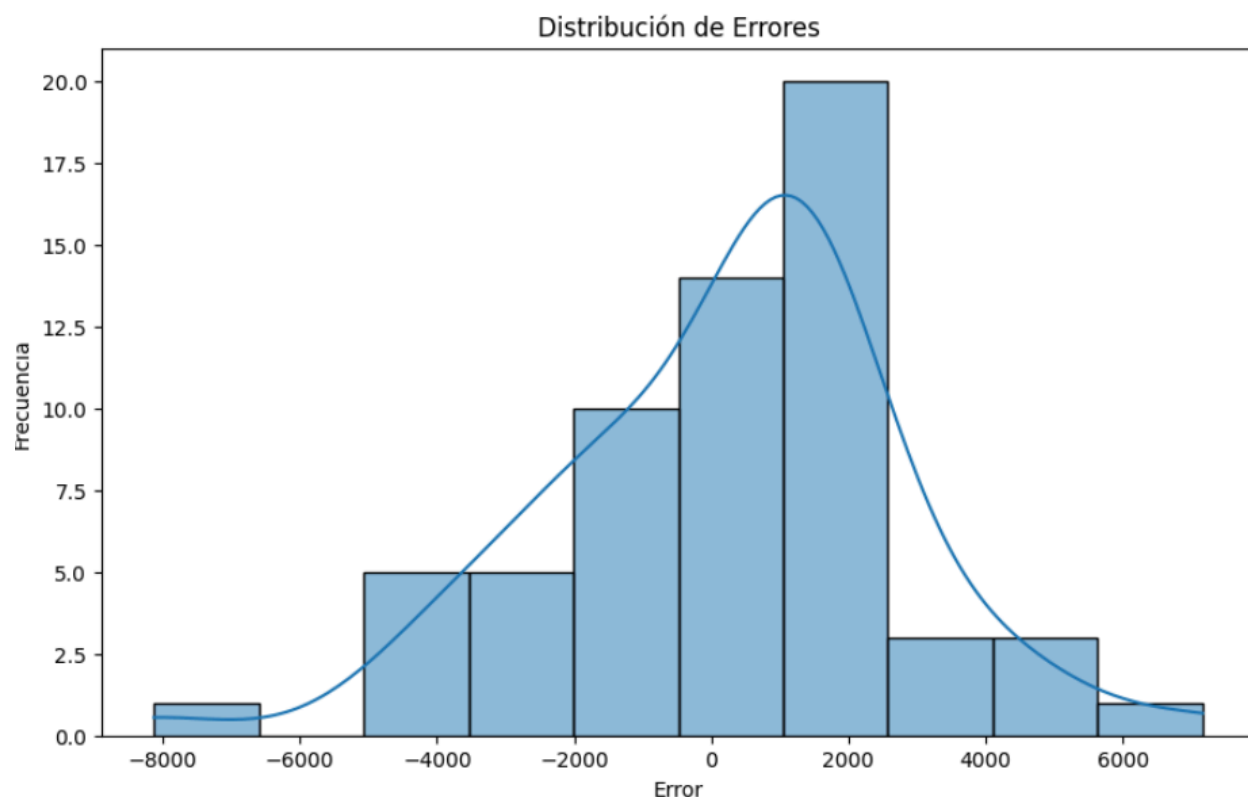


Modelo de Random Forest: Al igual que en los modelos anteriores, se observa una tendencia creciente. Sin embargo, ciertos puntos se desvían de la línea de referencia, lo que también sugiere que el modelo no tiene un ajuste perfecto.

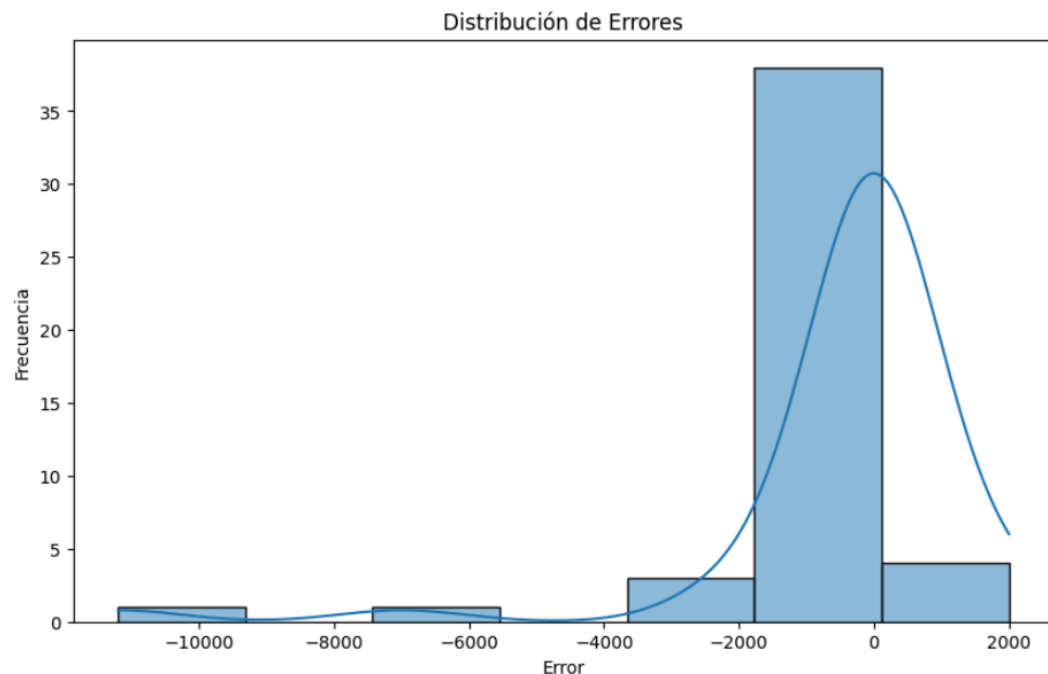


Evaluación a través de la Distribución de Errores

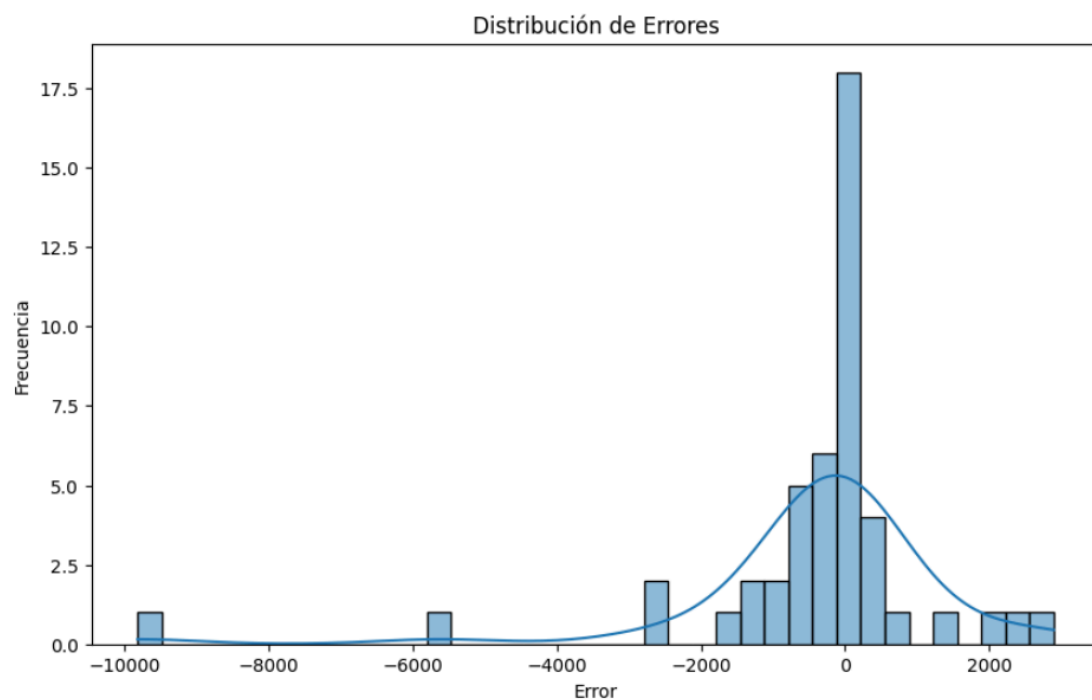
Modelo de Regresión Lineal: La gráfica muestra la distribución de los errores, donde la mayoría se concentran alrededor de cero con una forma acampanada. Esto sugiere un buen ajuste general del modelo. Aunque hay algunos errores extremos, su frecuencia es relativamente baja, indicando que el modelo no presenta un sesgo sistemático. Esta visualización es útil para identificar áreas de mejora.



Modelo de Árboles de Decisión: Similar al modelo de regresión lineal, la gráfica muestra que la mayoría de los errores se concentran alrededor de cero. A pesar de la presencia de algunos errores grandes, su frecuencia es baja, lo que indica un buen ajuste general y sin sesgo sistemático.

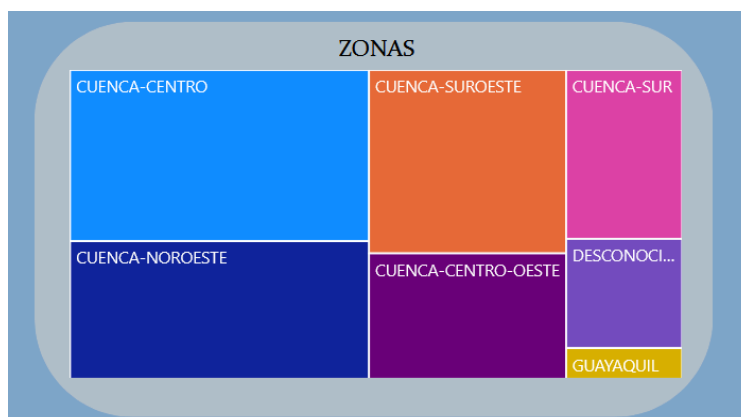


Modelo de Random Forest: La gráfica revela que la mayoría de los errores se concentran alrededor de cero, pero se observan errores extremos con una frecuencia relativamente alta. Esto sugiere que el modelo podría tener dificultades para predecir ciertos valores extremos. Esta visualización permite identificar áreas de mejora y evaluar la calidad general del modelo.

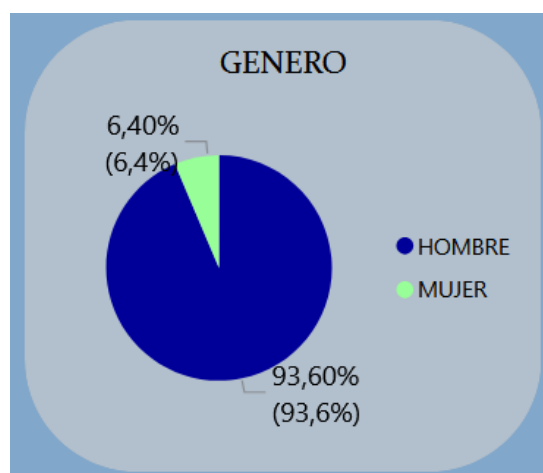


Resultados en Power BI

Las variables utilizadas para medir el nivel de satisfacción de los clientes son la frecuencia de compras y la cantidad adquirida. Los resultados indican que la zona con mayor satisfacción respecto a los productos adquiridos es la zona de Cuenca Norte.



Al analizar el género de nuestros clientes, se observa que el 83.17 % de los clientes satisfechos son hombres, mientras que el 16.83 % son mujeres.



8. CONCLUSIONES

A partir de los resultados obtenidos, se pueden extraer las siguientes conclusiones:

- Es fundamental contar con un dataset bien construido y estructurado que contenga todas las variables necesarias para el estudio o la generación del sistema. La limitación en la calidad y cantidad de datos puede restringir las alternativas disponibles para el entrenamiento del modelo.
- Suficiencia de datos: En este caso, al predecir la cantidad de stock mensual para cada uno de los productos, se identificó que la falta de registros suficientes para algunos productos fue una limitante. Esto resultó en errores menores en las predicciones de los modelos, lo que subraya la necesidad de ampliar la base de datos.
- Para que este desarrollo sea considerado un sistema utilizable, sería beneficioso integrar todas las herramientas generadas. Esto implica establecer una base de datos que alimente y entrene continuamente el modelo o modelos, permitiendo que los resultados se publiquen de manera automática en el cuadro de mando. De esta forma, el sistema se vuelve útil y puede generar beneficios significativos para MQ Distribuidora.
- La integración de Power BI y Google Colab en el contexto de una distribuidora de materiales de construcción permite optimizar la toma de decisiones estratégicas. Al analizar el dataset de ventas, inventarios y tendencias del mercado en Google Colab, se pueden identificar patrones y áreas de mejora que faciliten la planificación y ejecución de estrategias comerciales más efectivas.

9. BIBLIOGRAFÍA

- Flores Taquiri, G. A., & Montalvo Celis, J. S. (2024). Machine learning para la predicción en la gestión de inventario dirigida a PYMES de venta de productos tecnológicos. https://repositorio.ucv.edu.pe/bitstream/handle/20.500.12692/151376/Flores_TGA-Montalvo_CJS-SD.pdf?sequence=1&isAllowed=y
- Boada, A. J. (2017). Sistema de proyección de la demanda. Caso práctico de predicción automatizada en empresas de venta por catálogo. *Revista Perspectiva Empresarial*, 4(1), 23-41. <https://revistas.ceipa.edu.co/index.php/perspectiva-empresarial/article/view/122/55>
- Moina Álvarez, V. B., & Changoluisa Chillagana, J. L. (2021). Diseño e implementación de un prototipo para el control de gestión de inventario del producto terminado en la Fábrica de Cueros El AL-CE basado en inteligencia artificial. <http://dspace.esPOCH.edu.ec/bitstream/123456789/15965/1/85T00647.pdf>
- Berenguel Velasquez, J. E. (2019). Sistema web utilizando aprendizaje automático y análisis textual para detectar la satisfacción del cliente en la Empresa Mallhogar SAC. https://repositorio.ucv.edu.pe/bitstream/handle/20.500.12692/104422/Berenguel_VJE-SD.pdf?sequence=1&isAllowed=y
- Morales Tabares, Z. E. (2016). Modelo multivariado de predicción del stock de piezas de repuesto para equipos médicos. <https://repositorio.uci.cu/handle/123456789/7157>
- Cortina, V. G. (2015). Aplicación de la metodología crisp-dm a un proyecto de minería de datos en el entorno universitario. Trabajo de Fin de Grado (Universidad Carlos III de Madrid). https://d1wqtxts1xzle7.cloudfront.net/49071533/PFC_Victor_Galan_Cortina_DM-libre.pdf?1474668369=&response-content-disposition=inline%3B+filename%3DPROYECTO_FIN_DE_CARRERA_APLICACION_DE_LA.pdf&Expires=1739335122&Signature=Jmxg33wkf-izQk-qHJdKkpr4Zwq2xFIY2ufKhLwl-r8U6Lq5kSM5x6tBhb4P8Ou~WfZZdV7Bo3CFAYBJql3fX8KiNZuuf-clkDMBsqXLHoPKDNNLb38PmV0Hjxn7B~oB7~MaKXFCbmriOfafB2xlvN95f5pSIsylAvtzloSoBzps~J0Wq7pjxhot9P2IY2Ei7Jf6-c7rzJUWUImeZGooL7Bpi2XO64vvEjG2EYyjNTVzGCKmpm6bcP9d7C7RoY~58GWbTPZysHdRg-xY7IO-DwyWJzv5pmfXOHKiNZHdAXgZhfvf5NaOLNCb2oXptVWhKAKsfMSZXIhsFhp~RGex4g__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA
- SARA, V. N. (2016). LA METODOLOGÍA CRISP-DM. <http://ri.uaemex.mx/bitstream/handle/20.500.11799/64111/secme-12408.pdf?sequence=1>

- Moine, J. M., Haedo, A. S., & Gordillo, S. E. (2011). Estudio comparativo de metodologías para minería de datos. In XIII Workshop de Investigadores en Ciencias de la Computación.
https://sedici.unlp.edu.ar/bitstream/handle/10915/20034/Documento_completo.pdf?sequence=1&isAllowed=y
- Bermúdez León, M. J. (2020). Proceso de diseño.
<https://repositorio.usam.ac.cr/xmlui/bitstream/handle/11506/2123/LEC%20ING%20SIST%200067%202020.pdf?sequence=1&isAllowed=y>
- Gurtubay Regulez, A. (2021). Proyecto de optimización de los servicios y ventas mediante Machine Learning.
- Gaona, A. L. (2012). El modelo Entidad-Relación.
https://d1wqtxts1xzle7.cloudfront.net/53070088/er-libre.pdf?1494426926=&response-content-disposition=inline%3B+filename%3DEl_modelo_Entidad_Relacion.pdf&Expires=1739422312&Signature=UJrQPfjKupbYgDXskjhgh~0ZkyFuLfBbYyEZht9xWW61tLYQ0CoDNNfhZAqLKgx21RcmV~wXLI7vk4D4IYYgbmm2fen8tAH0d-OdNxICjiw-PswOu7ZX4mQY1VNhvoFYSIN3LbIKWLxGOuk6JCULhLCXNh9yCI3KRmLjEPnT7ZBS0TiyXRNLJks5zwRWsdX74WSE4PYO5CFZKYPrZxu-lQ11e9r~FbJlgRz9cNJ57hcsa8luHr3bpl8NCT~hM4XgzVwdBd3YrVFF~GTCyWOZ9OjIGg8BN4tdMln~wr~t2xFJZ37~J7ERHpSdnU6LwGDbekdMVPTTgm0DZQV3lq0Q__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA
- Li, H., & Zuo, J. (2019). Application of machine learning in construction project management. *Automation in Construction*, 102, 1-10.
- Elmasri, R., Navathe, S. B., Castillo, V. C., Pérez, G. Z., & Espiga, B. G. (2007). *Fundamentos de sistemas de bases de datos*. Pearson educación.
- Piñeiro Gómez, J. O. (2024). *Diseño de bases de datos relacionales*. Ediciones Paraninfo, SA.
https://books.google.es/books?hl=es&lr=lang_es&id=pas0EQAAQBAJ&oi=fnd&pg=PR5&dq=Diseno+de+el+modelo+Relacional+de+base+de+datos&ots=SddlefcOLw&sig=XWM6d4hVGm5aLk6_p1rVASFSJ1c#v=onepage&q=Diseno%20de%20el%20modelo%20Relacional%20de%20base%20de%20datos&f=false
- Adrián, T. E. (2016). Proceso de normalización de base de datos en un modelo relacional.
<https://core.ac.uk/download/pdf/80532387.pdf>
- Escofet, C. M. (2002). *El lenguaje SQL*. UOC, la universidad virtual.
<https://pdfcursos.com/pdf/0018-lenguaje-sql.pdf>
- Ginestà, M. G., & Mora, O. P. (2012). *Bases de datos en PostgreSQL*. SI]:[sn].
<https://artpatcusco.com/sis/pdf/20140527172742en.pdf>

- Rodríguez Rodríguez, J. E. (2010). Fundamentos de minería de datos. Editorial Universidad Distrital Francisco José de Caldas. Editorial UD.
https://books.google.es/books?hl=es&lr=lang_es&id=t7zvEAAQBAJ&oi=fnd&pg=PT7&dq=preprocesamiento+de+datos+en+minería+de+datos&ots=8B4fr30eET&sig=2hPBYZTh_ufexTuZzx1c3ptlvBw#v=onepage&q=preprocesamiento%20de%20datos%20en%20minería%20de%20datos&f=false
- Rodríguez Rodríguez, J. E. (2010). Fundamentos de minería de datos. Editorial Universidad Distrital Francisco José de Caldas. Editorial UD.
https://books.google.es/books?hl=es&lr=lang_es&id=t7zvEAAQBAJ&oi=fnd&pg=PT7&dq=T%C3%A9cnicas+de+Procesamiento+y+Preprocesamiento+de+Datos+en+Minería%20de+Datos&ots=8B4fr8WbDQ&sig=DRnzZ9yTM-YqAWc-4JfoHWyAvRo#v=onepage&q=T%C3%A9cnicas%20de%20Procesamiento%20y%20Preprocesamiento%20de%20Datos%20en%20Minería%20de+Datos&f=false
- Dagnino, J. (2014). Regresión lineal. Rev. chil. anest, 43(2).
https://www.sachile.cl/upfiles/revistas/54e63943b5d69_14_regresion-2-2014_edit.pdf
- Cajahuanca, J. E. V., Raymundo, Á. F. N., Franco, A. C. L., & Flores, J. D. J. (2022). Deserción universitaria: Evaluación de diferentes algoritmos de Machine Learning para su predicción. Revista de ciencias sociales, 28(3), 362-375.
- Arana, C. (2021). Modelos de aprendizaje automático mediante árboles de decisión (No. 778). Serie Documentos de Trabajo.
<https://www.econstor.eu/bitstream/10419/238403/1/778.pdf>
- Espinosa-Zúñiga, J. J. (2020). Aplicación de algoritmos Random Forest y XGBoost en una base de solicitudes de tarjetas de crédito. Ingeniería, investigación y tecnología, 21(3).
https://www.scielo.org.mx/scielo.php?pid=S1405-77432020000300002&script=sci_arttext