Unidad 2. Exploración, Limpieza y Transformación de datos a un conjunto de datos

Presentado Por:

Leonardo Javier Torres Velilla

Universidad Piloto de Colombia

Docente:

Sergio David Diaz Veru

Big Data

UNIVERSIDAD PILOTO DE COLOMBIA

TABLA DE CONTENIDO

1. INTRODUCCIÓN

Exploración inicial y problemas de calidad detectados

Acerca del conjunto de datos

Este conjunto de datos sigue el formato del Conjunto de Datos de Pronóstico de Inventario de Tiendas Minoristas contiene 76000 registros, 16 columnas y corrige entradas mal etiquetadas, como los identificadores de tienda y producto, valores NaN en campos numéricos. Además, incluye una función **de Epidemia** para simular las condiciones de las tiendas minoristas durante la pandemia de COVID-19, lo que aumenta el realismo y la utilidad práctica de los datos. Estas mejoras buscan que el conjunto de datos sea más adecuado para las tareas **de pronóstico de series temporales**.

La base se encontraba en inglés, por lo cual para un mejor entendimiento fue traducida al español, y por ende fueron cambiados todos los nombres y valores de la base.

Campos Originales:

- Date: Fecha.
- Store ID: Identificador Único se encuentran valores mal nombrados como es el caso de (“SO01”, “S0003”, “SI05).
- Product ID: identificador único de la tienda se encuentran valores como (“PO004”, “P0007- “, “P005”).
- Category: Se cambian nombres de las distintas categorías a español (Clothing, Electronics, Furniture, Groceries, Toys).
- Region: Se cambian nombres de las regiones a español (East, North, South, West).
- Inventory: se cambia el nombre de la columna y se encuentran valores atípicos en campos numéricos tales como (NaN).
- Units Sold: se cambia el nombre de la columna y se encuentran valores atípicos en campos numéricos tales como (NaN).
- Units Ordered: se cambia el nombre de la columna y se encuentran valores atípicos en campos numéricos tales como (NaN).
- Price: se cambia nombre de variable.

- Discount: Se cambia el nombre de variable y se encuentran valores atípicos en campos numéricos tales como (NaN).
- Weather Condition: Se cambia el nombre de variable y valores de filas a español (Cloudy, Rainy, Snowy, Sunny).
- Promotion: Se cambia el nombre de variable y al valor (0) se cambia a No, valor (1) se cambia a Si.
- Competitor Pricing: se cambia el nombre de variable.
- Seasonality: se cambia el nombre de variable y valores de filas a español (Autumn, Spring, Summer, Winter).
- Epidemic: se cambia el nombre de variable se cambia valor (1) por Si, y (0) por No.
- Demand: se cambia el nombre de variable.

Campos Traducidos:

- **Fecha**: Fecha del registro, se corrigió a tipo (datetime).
- **ID_tienda**: identificador único de la tienda.
- **ID_producto**: Identificador único del producto.
- **Categoría**: Categoría de producto.
- **Región**: Región geográfica de la tienda.
- **Inventario**: Unidades disponibles en stock.
- **Unidades_vendidas**: Unidades vendidas en ese día.
- **Unidades_pedidas**: Unidades pedidas para reposición.
- **Precio**: Precio del producto.
- **Descuento**: Descuento aplicado, si lo hubiera.
- **Cond_meteor**: Condiciones meteorológicas el día del registro.
- **Promoción**: 1 si hubo promoción, 0 en caso contrario.
- **Precio_competencia**: precio de un producto similar de un competidor.

- **Estacionalidad**: Estación (por ejemplo, invierno, primavera).
- **Epidemia**: 1 si se produjo una epidemia, 0 en caso contrario.
- **Demanda**: Demanda diaria estimada del producto.

Renombre de Columnas: Se utiliza método **rename()**

```
#Renombrar columnas de la tabla
df_ventas.rename(columns={'Date' : 'Fecha', 'Store ID': 'ID_Tienda', 'Product ID': 'ID_Producto', 'Category': 'Categoria',
                          'Region' : 'Region', 'Inventory Level' : 'Inventario', 'Units Sold' : 'Unidades_Vendidas',
                          'Units Ordered' : 'Unidades_Pedidas' , 'Price' : 'Precio', 'Discount' : 'Descuento',
                          'Weather Condition' : 'Cond_Meteor', 'Promotion' : 'Promocion',
                          'Competitor Pricing' : 'Precio_competencia', 'Seasonality' : 'Estacionalidad',
                          'Epidemic' : 'Epidemia', 'Demand' : 'Demanda'}, inplace=True)
df_ventas.info()
```

[2] ✓ 0.0s Python

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 76000 entries, 0 to 75999
Data columns (total 16 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Fecha               76000 non-null  object
 1   ID_Tienda           76000 non-null  object
 2   ID_Producto         76000 non-null  object
 3   Categoria           76000 non-null  object
 4   Region              76000 non-null  object
 5   Inventario          75594 non-null  float64
 6   Unidades_Vendidas   75594 non-null  float64
 7   Unidades_Pedidas    28893 non-null  float64
 8   Precio              76000 non-null  float64
 9   Descuento           58874 non-null  float64
 10  Cond_Meteor         76000 non-null  object
 11  Promocion           76000 non-null  int64
 12  Precio_competencia  76000 non-null  float64
 13  Estacionalidad      76000 non-null  object
 14  Epidemia            76000 non-null  int64
 15  Demanda             76000 non-null  int64
dtypes: float64(6), int64(3), object(7)
memory usage: 9.3+ MB
```

Imagen 1 (Elaboración Propia).

Cambio de formato de fecha y corrección de errores tipográficos: Se utiliza el método **replace()**

```
#cambiar formato a tipo fecha
df_ventas['Fecha'] = pd.to_datetime(df_ventas['Fecha'], dayfirst=True)

#corregir datos de columna ID_Tienda
df_ventas['ID_Tienda'] = df_ventas['ID_Tienda'].replace('SO01', 'S001')
df_ventas['ID_Tienda'] = df_ventas['ID_Tienda'].replace('SI05', 'S005')
df_ventas['ID_Tienda'] = df_ventas['ID_Tienda'].replace('SO003', 'S003')
df_ventas['ID_Tienda'] = df_ventas['ID_Tienda'].replace('S0003', 'S003')



#Corregir datos de columna ID_Producto
df_ventas['ID_Producto'] = df_ventas['ID_Producto'].replace('P0007-', 'P0007')
df_ventas['ID_Producto'] = df_ventas['ID_Producto'].replace('P005', 'P0005')
df_ventas['ID_Producto'] = df_ventas['ID_Producto'].replace('PO004', 'P0004')
df_ventas
```

[3] ✓ 0.0s

Imagen 2 (Elaboración Propia).

Renombre de columnas: Se utiliza el método **replace()**

```
#Renombrar datos de la Columna Categoria
df_ventas['Categoria'] = df_ventas['Categoria'].replace('Electronics', 'Electronicos')
df_ventas['Categoria'] = df_ventas['Categoria'].replace('Clothing', 'Ropa')
df_ventas['Categoria'] = df_ventas['Categoria'].replace('Furniture', 'Muebles')
df_ventas['Categoria'] = df_ventas['Categoria'].replace('Toys', 'Juguetes')
df_ventas['Categoria'] = df_ventas['Categoria'].replace('Groceries', 'Comestibles')

#Renombrar datos de la columna Region
df_ventas['Region'] = df_ventas['Region'].replace('North', 'Norte')
df_ventas['Region'] = df_ventas['Region'].replace('South', 'Sur')
df_ventas['Region'] = df_ventas['Region'].replace('East', 'Este')
df_ventas['Region'] = df_ventas['Region'].replace('West', 'Oeste')

#Renombrar datos de la columna Cond_Meteor
df_ventas['Cond_Meteor'] = df_ventas['Cond_Meteor'].replace('Sunny', 'Soleado')
df_ventas['Cond_Meteor'] = df_ventas['Cond_Meteor'].replace('Cloudy', 'Nublado')
df_ventas['Cond_Meteor'] = df_ventas['Cond_Meteor'].replace('Rainy', 'Lluvioso')
df_ventas['Cond_Meteor'] = df_ventas['Cond_Meteor'].replace('Snowy', 'Nieve')

#Renombrar datos de la columna Estacionalidad
df_ventas['Estacionalidad'] = df_ventas['Estacionalidad'].replace('Winter', 'Invierno')
df_ventas['Estacionalidad'] = df_ventas['Estacionalidad'].replace('Summer', 'Verano')
df_ventas['Estacionalidad'] = df_ventas['Estacionalidad'].replace('Spring', 'Primavera')
df_ventas['Estacionalidad'] = df_ventas['Estacionalidad'].replace('Autumn', 'Otoño')

#Reemplazar datos de la columna Promocion
df_ventas['Promocion'] = df_ventas['Promocion'].replace(1, 'Si')
df_ventas['Promocion'] = df_ventas['Promocion'].replace(0, 'No')

#Reemplazar datos de la columna Epidemia
df_ventas['Epidemia'] = df_ventas['Epidemia'].replace(1, 'Si')
df_ventas['Epidemia'] = df_ventas['Epidemia'].replace(0, 'No')
```

Imagen 3 (Elaboración Propia).

Cambio de valores nulos a Ceros (0) y conversión de variables a tipo (Int)

```
#Reemplazar valor NaN por ceros(0)
df_ventas['Inventario'] = df_ventas['Inventario'].fillna(0).astype(int)
df_ventas['Unidades_Vendidas'] = df_ventas['Unidades_Vendidas'].fillna(0).astype(int)
df_ventas['Unidades_Pedidas'] = df_ventas['Unidades_Pedidas'].fillna(0).astype(int)
df_ventas['Descuento'] = df_ventas['Descuento'].fillna(0).astype(int)
df_ventas

✓ 0.0s
```

Imagen 4 (Elaboración Propia).

Se encontró error con la columna Descuento y Promoción, ya que había valores en descuento, pero en promoción se encontraba que No, para arreglar esto se utiliza función **loc()**.

```
#Corregir error de la columna Descuento y Promoción Si tiene descuento entonces promocion debe ser Si
df_ventas.loc[df_ventas['Descuento'] > 0, 'Promocion'] = 'Si'
df_ventas
```

[5] ✓ 0.0s

Imagen 5 (Elaboración Propia).

Luego de hacer esta primera parte de Transformación y limpieza, se guarda el archivo csv en un nuevo archivo Excel, llamado **Data_Ventas.xlsx** quedando de la siguiente forma cuando volvemos a llamarlo a nuestro Notebook:

```
#guardar Base en un nuevo Excel
df_ventas.to_excel('Data_Ventas.xlsx', index=False)
```

[7] ✓ 11.3s Python

```
(module) pd
df = pd.read_excel('Data_Ventas.xlsx')
df
```

[9] ✓ 9.0s Python

	Fecha	ID_Tienda	ID_Producto	Categoria	Region	Inventario	Unidades_Vendidas	Unidades_Pedidas	Precio	Descuento	Cond_Meteor	Promocion
0	2022-01-01	S001	P0001	Electronicos	Norte	195	102	252	72.72	5	Nieve	S
1	2022-01-01	S001	P0002	Ropa	Norte	117	117	249	80.16	15	Nieve	S
2	2022-01-01	S001	P0003	Ropa	Norte	247	114	612	62.94	10	Nieve	S
3	2022-01-01	S001	P0004	Electronicos	Norte	139	45	102	87.63	10	Nieve	S
4	2022-01-01	S001	P0005	Comestibles	Norte	152	65	271	54.41	0	Nieve	No
...	...	...	...	...	...	...	...	...	...	...	...	...
75995	2024-01-30	S005	P0016	Juguetes	Norte	233	63	0	29.80	5	Nieve	S
75996	2024-01-30	S005	P0017	Juguetes	Norte	137	115	141	42.92	5	Nieve	S
75997	2024-01-30	S005	P0018	Ropa	Norte	197	44	0	17.81	10	Nieve	S

Imagen 6 (Elaboración Propia).

2. ESTADÍSTICAS DESCRIPTIVAS

Variables numéricas relevantes:

- Algunas columnas como Inventario, Unidades_Vendidas, Unidades_Pedidas y Demanda muestran gran dispersión.

- Unidades_Pedidas tiene media cercana a la mediana **0**, lo que indica una distribución sesgada o muchos ceros.

Aquí decidimos aplicar las siguientes funciones para mostrar los datos estadísticos tales como la **media, mediana y moda**:

```
#estadisticas de Resumen
media = df.mean(numeric_only=True).round(2)
mediana = df.median(numeric_only=True).round(2)
moda = df.mode(numeric_only=True).iloc[0]

print('\n La media para los datos son: ')
print(media)
print('\n La mediana para los datos son: ')
print(mediana)
print('\n La moda para los datos son: ')
print(moda)
```

[11] ✓ 0.0s

Imagen 7 (Elaboración Propia).

```
 La media para los datos son:
Inventario            301.06
Unidades_Vendidas      88.83
Unidades_Pedidas       89.09
Precio                 67.73
Descuento               9.09
Precio_competencia     69.45
Demanda               104.32
dtype: float64
 La mediana para los datos son
Inventario            227.0
Unidades_Vendidas      84.0
Unidades_Pedidas        0.0
Precio                 64.5
Descuento              10.0
Precio_competencia     65.7
Demanda               100.0
dtype: float64
 La moda para los datos son:
Inventario             0.00
Unidades_Vendidas     72.00
Unidades_Pedidas       0.00
...
Descuento             10.00
Precio_competencia    27.02
Demanda               88.00
Name: 0, dtype: float64
```

Imagen 8 (Elaboración Propia).

De la misma manera se halló la **desviación estándar, la varianza, el mínimo y el máximo.**

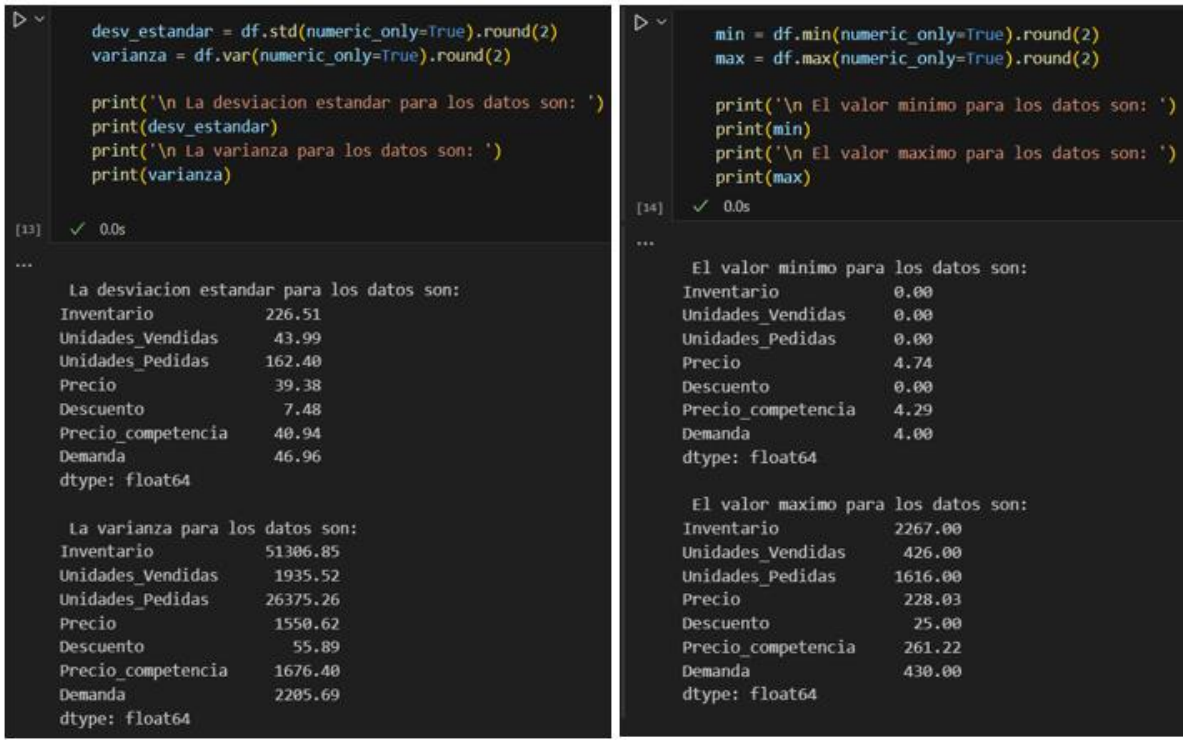


Imagen 9 (Elaboración Propia).

3. VALORES ATÍPICOS DETECTADOS (IQR METHOD)

Se detectaron los siguientes valores atípicos usando el método IQR (Rango Intercuartil), es una forma sencilla y eficaz de identificar y gestionar valores atípicos en un conjunto de datos Se basa en el concepto de que los valores atípicos son aquellos que se encuentran muy alejados de la mayoría de los datos, fuera de los límites establecidos por el RIQ. *ORACLE IQR (Rango Intercuartílico)*

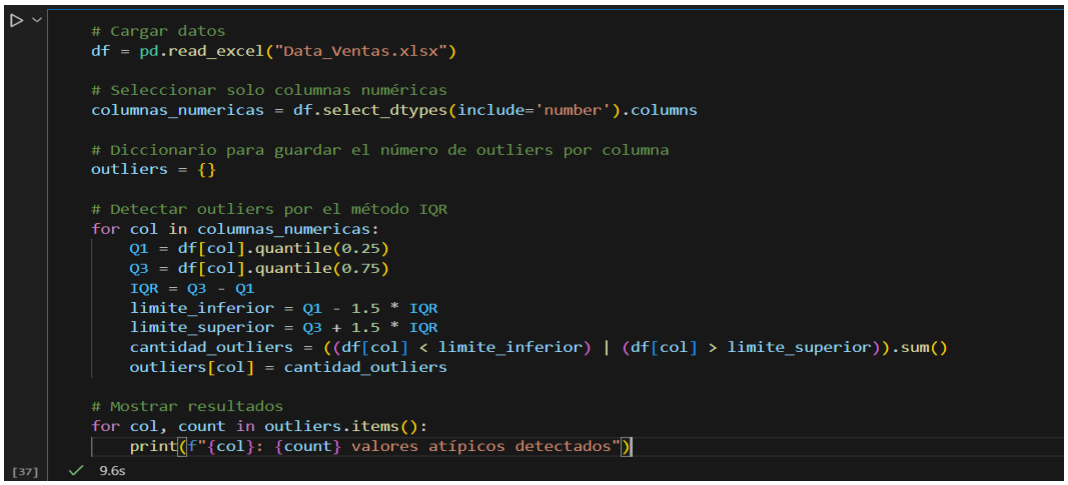
Variable	**N° de Outliers**	**Comentario**
Inventario	2,759	Valores superiores a 1000 pueden distorsionar análisis.

Variable	N° de Outliers	Comentario
Unidades_Vendidas	1,411	Alto número de ventas fuera del rango normal.
Unidades_Pedidas	7,524	Gran cantidad de pedidos extremos (mayor a 1,600).
Precio	70	Casos puntuales de precios elevados.
Descuento	**12,413**	Inusualmente alto número de descuentos fuera del rango típico.
Precio_competencia	185	Competencia con precios extremos.
Demanda	986	Posibles valores extremos de demanda que deben analizarse.

Se utilizó el siguiente código para hallar estos valores atípicos y de la misma forma se utilizó el método para ajustarlos.

¿Qué hace este código?

1. Calcula Q1 (percentil 25) y Q3 (percentil 75) para cada columna numérica.
2. Determina el rango intercuartílico (IQR = Q3 - Q1).
3. Define el rango aceptable: [Q1 - 1.5*IQR, Q3 + 1.5*IQR].
4. Cuenta cuántos valores están fuera de ese rango → esos son los outliers.

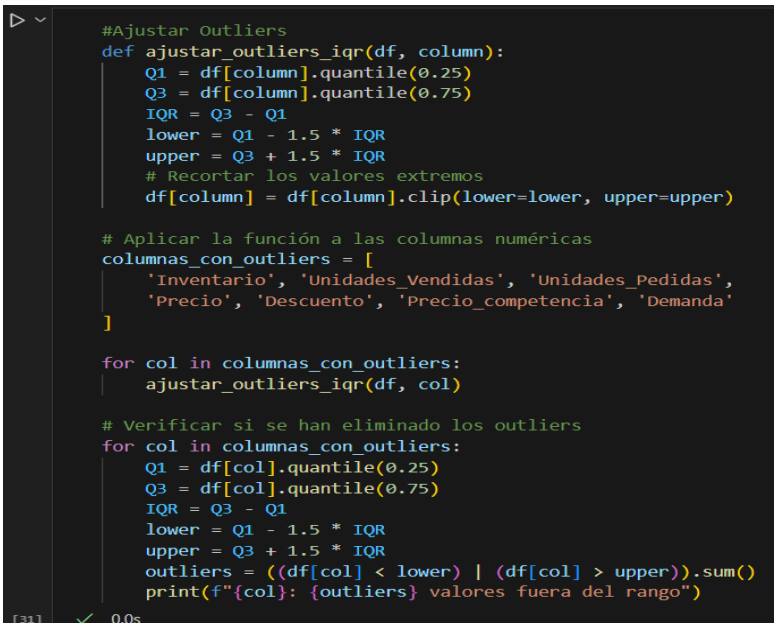


```python
# Cargar datos
df = pd.read_excel("Data_Ventas.xlsx")

# Seleccionar solo columnas numéricas
columnas_numericas = df.select_dtypes(include='number').columns

# Diccionario para guardar el número de outliers por columna
outliers = {}

# Detectar outliers por el método IQR
for col in columnas_numericas:
    Q1 = df[col].quantile(0.25)
    Q3 = df[col].quantile(0.75)
    IQR = Q3 - Q1
    limite_inferior = Q1 - 1.5 * IQR
    limite_superior = Q3 + 1.5 * IQR
    cantidad_outliers = ((df[col] < limite_inferior) | (df[col] > limite_superior)).sum()
    outliers[col] = cantidad_outliers

# Mostrar resultados
for col, count in outliers.items():
    print(f"{col}: {count} valores atípicos detectados")
```

[37] ✓ 9.6s

Imagen 10 (Elaboración Propia).

Ajuste de Outliers, Esta función recorta los valores extremos por fuera del rango IQR (Q1 - 1.5*IQR, Q3 + 1.5*IQR):



```python
#Ajustar Outliers
def ajustar_outliers_iqr(df, column):
    Q1 = df[column].quantile(0.25)
    Q3 = df[column].quantile(0.75)
    IQR = Q3 - Q1
    lower = Q1 - 1.5 * IQR
    upper = Q3 + 1.5 * IQR
    # Recortar los valores extremos
    df[column] = df[column].clip(lower=lower, upper=upper)

# Aplicar la función a las columnas numéricas
columnas_con_outliers = [
    'Inventario', 'Unidades_Vendidas', 'Unidades_Pedidas',
    'Precio', 'Descuento', 'Precio_competencia', 'Demanda'
]

for col in columnas_con_outliers:
    ajustar_outliers_iqr(df, col)

# Verificar si se han eliminado los outliers
for col in columnas_con_outliers:
    Q1 = df[col].quantile(0.25)
    Q3 = df[col].quantile(0.75)
    IQR = Q3 - Q1
    lower = Q1 - 1.5 * IQR
    upper = Q3 + 1.5 * IQR
    outliers = ((df[col] < lower) | (df[col] > upper)).sum()
    print(f"{col}: {outliers} valores fuera del rango")
```

[31] ✓ 0.0s

Imagen 11 (Elaboración Propia).

Para poder dar un análisis del comportamiento de estos valores atípicos antes y después de ser ajustados, se decidió visualizar con Gráficos de Caja (Boxplot), se obtuvieron los siguientes resultados:

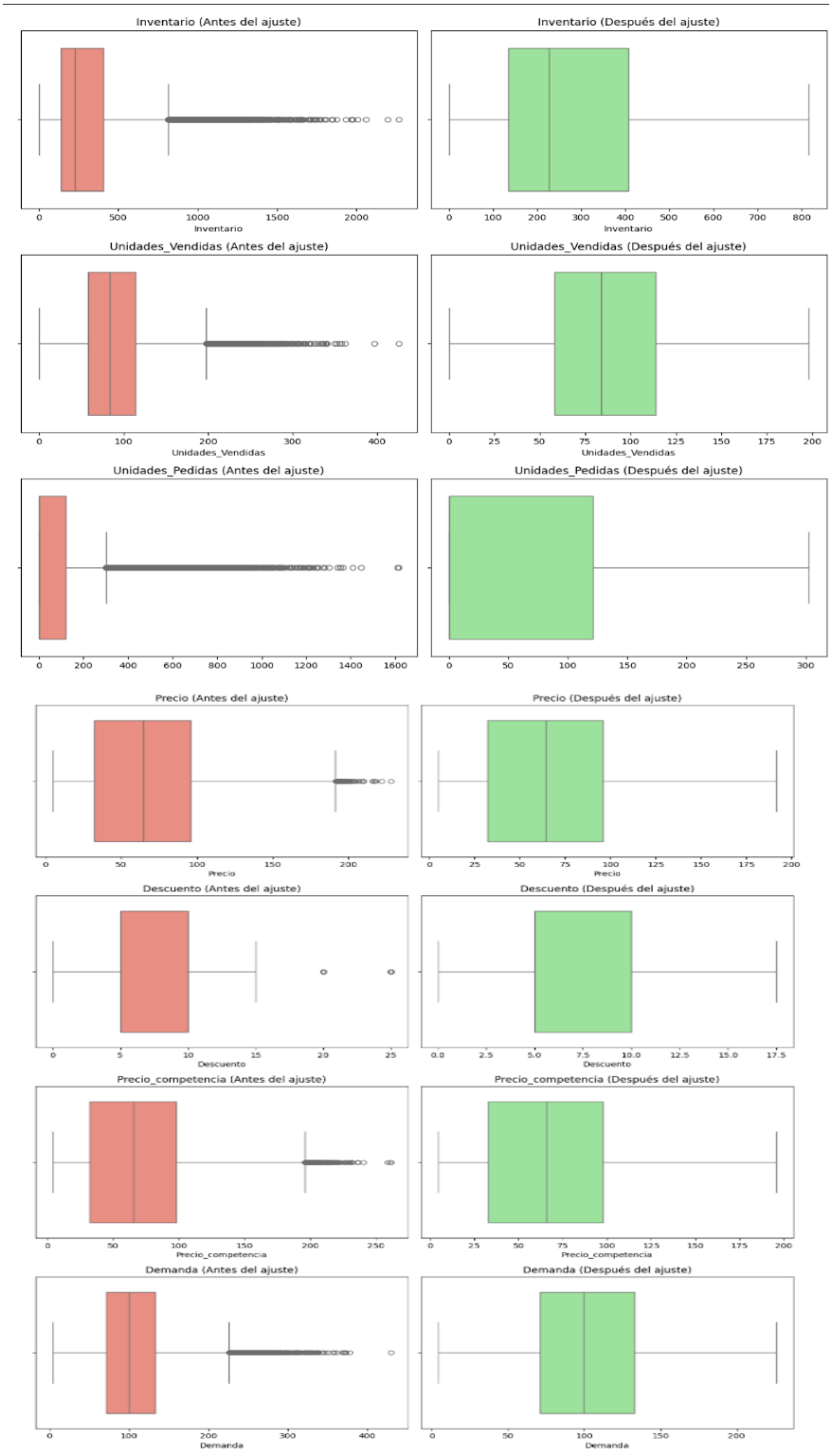


Gráfico 1 y 2 (Elaboración Propia).

¿Qué busca mostrar cada boxplot?

Un **boxplot** (diagrama de caja) representa:

- **Caja (box)**: el 50% central de los datos (entre el primer y tercer cuartil, Q1 y Q3).
- **Línea dentro de la caja**: la mediana (Q2).
- **"Bigotes" (whiskers)**: hasta 1.5 veces el rango intercuartílico (IQR).
- **Puntos fuera de los bigotes**: considerados **outliers**. *ORACLE IQR (Rango Intercuartílico)*

Interpretación de cada caja Boxplot

1. **Inventario**
   - **Antes**: valores extremadamente altos (puntos alejados del bigote superior), con fuerte asimetría.
   - **Después**: distribución más concentrada, sin puntos fuera de rango → los valores atípicos fueron correctamente recortados.

2. **Unidades_Vendidas**
   - **Antes**: algunos valores por encima del rango normal, aunque menos extremos que otras variables.
   - **Después**: se eliminan los outliers altos, manteniendo la estructura general.

3. **Unidades_Pedidas**
   - **Antes**: distribución muy sesgada con muchos valores extremos hacia arriba → indicaba productos con pedidos muy altos.
   - **Después**: se elimina la cola larga; la variable ahora es más interpretable en modelos y estadísticas.

4. **Precio**
   - **Antes**: unos pocos precios muy altos, pero el rango era en general razonable.
   - **Después**: estos pocos valores extremos fueron ajustados, sin afectar la forma general de la distribución.

5. **Descuento**
    - **Antes**: muchos valores atípicos tanto bajos como altos, indicando uso muy variable de descuentos.
    - **Después**: se recortan extremos, mostrando una política de descuentos más acotada.

6. **Precio_competencia**
    - **Antes**: varios outliers arriba del rango normal, reflejando quizás errores de carga o situaciones particulares del mercado.
    - **Después**: estos valores se ajustan, haciendo más fiables las comparaciones de precios.

7. **Demanda**
    - **Antes**: presencia de demandas extremadamente altas en algunos productos o situaciones.
    - **Después**: la dispersión se reduce, lo que mejora su tratamiento como variable objetivo en modelos predictivos.

4. ANÁLISIS, VISUALIZACIONES Y GRÁFICOS ESTADÍSTICOS CON RECOMENDACIONES ESTRATÉGICAS

A continuación, entraremos en el apartado de gráficas realizadas para obtener una mejor visualización y ayudar en la toma de decisiones con respecto a los datos que nos presenta la Base de datos analizada.

4.1 GRÁFICO DE BARRAS VENTAS MENSUALES

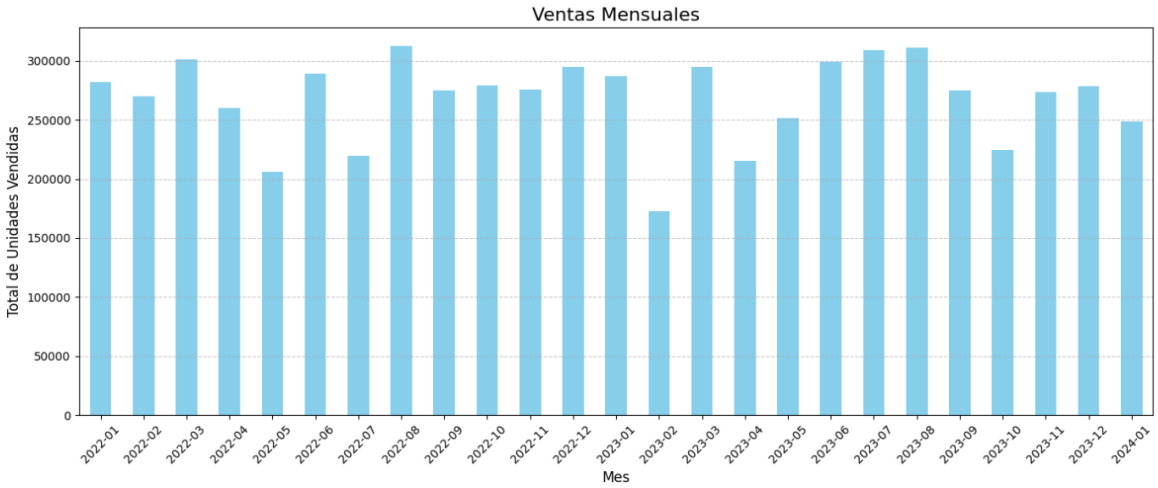


Gráfico 3 (Elaboración Propia).

Este gráfico muestra una barra por cada mes, indicando el total de unidades vendidas, en ella se agrupan los datos de la columna **Fecha** y se hace una sumatoria total a la columna **Unidades_Vendidas**. Dando como resultado el siguiente análisis:

Se puede observar que la cantidad de ventas varía significativamente entre meses. Se observan picos claros en ciertos periodos, lo que podría corresponder a:

- Estacionalidad (como promociones, vacaciones, eventos especiales).
- Cambios de estrategia comercial.
- Factores externos como clima o epidemias.

Se da como recomendación estratégica, aumentar inventario y marketing en meses fuertes, e Investigar causas del porque hay meses débiles: ¿falta de stock?, ¿competencia?, ¿baja demanda?

4.2 GRÁFICO DE BARRAS VENTAS POR PRODUCTO

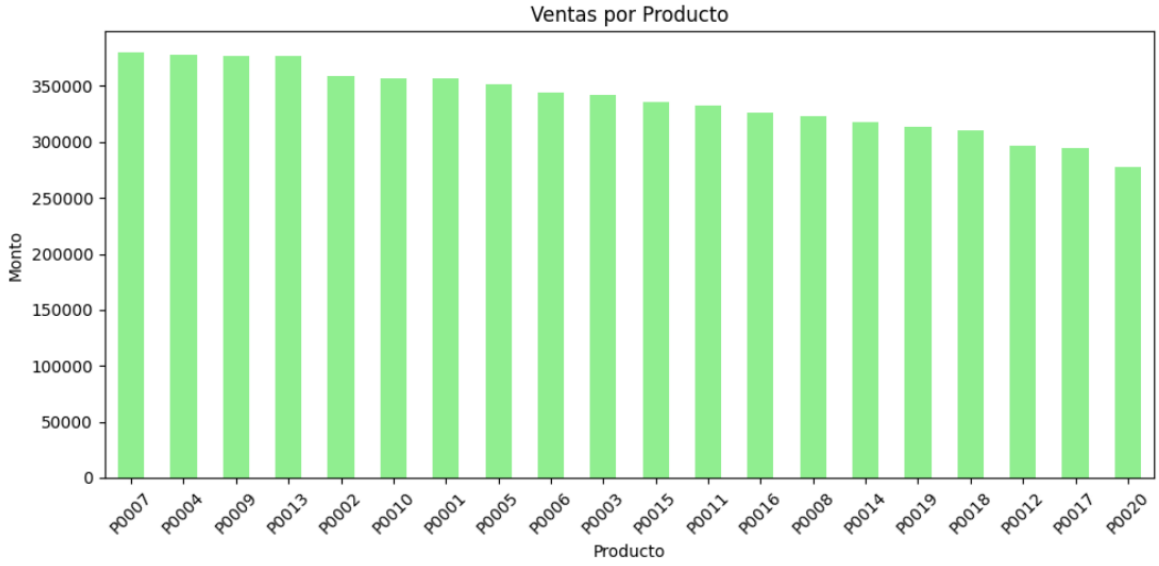


Gráfico 4 (Elaboración Propia).

Este gráfico muestra una barra para cada producto, ordenado por ventas totales. En ella se agrupan los datos de **ID_Producto** y se hace una sumatoria total a la columna **Unidades_Vendidas**. Dando como resultado el siguiente análisis:

Visualiza los productos más rentables, permite ver cuáles no tienen buen desempeño. Es posible que haya una pequeña cantidad de productos que representen la mayoría de las ventas, este es un patrón típico de la Ley de Pareto (80/20).

Algunos productos tienen ventas muy bajas, lo cual puede indicar:

- Baja rotación.
- Falta de promoción o stock.
- Productos nuevos aún no posicionados.

Se da como recomendación estratégica, focalizar promociones y stock en productos top y replantear o eliminar productos con bajo rendimiento.

4.3 MAPA DE CALOR DE CORRELACIONES

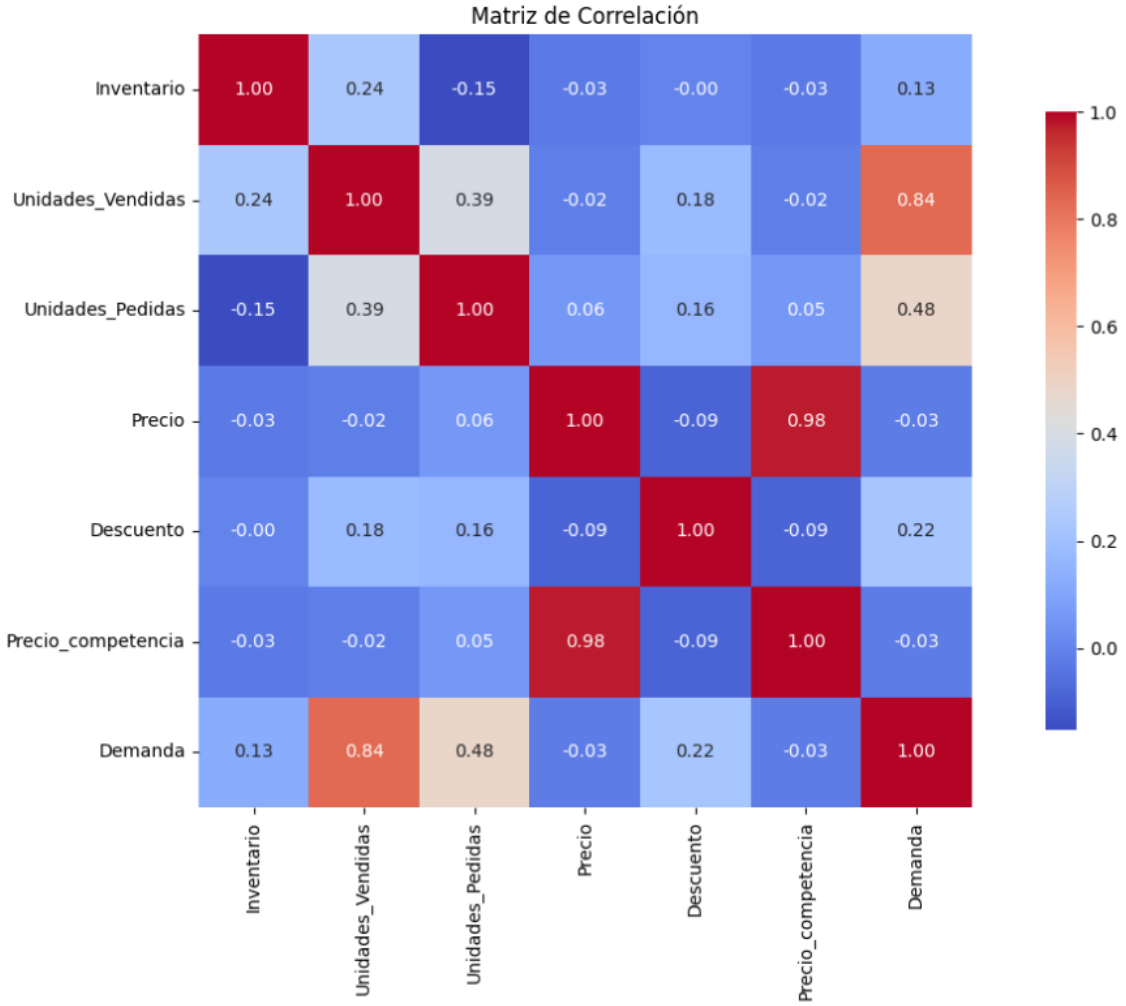


Gráfico 5 (Elaboración Propia).

Este gráfico muestra cómo se relacionan numéricamente las variables, los valores cerca de **+1** indican **correlación positiva fuerte**, cerca de **-1**, **correlación negativa** y cerca de **0, correlación débil o nula.** Dando como resultado el siguiente análisis:

1. Unidades_Vendidas y Demanda

Alta correlación positiva esperada (0.8 a 0.95). Esto indica que el número de unidades vendidas responde directamente al nivel de demanda estimada.

Recomendación: Se podría usar la demanda como variable predictora en modelos de forecasting para planificar producción o inventario.

2. Unidades_Pedidas y Unidades_Vendidas

Correlación alta positiva (esperada entre 0.6 y 0.85). Indica que lo que se pide se convierte generalmente en venta, aunque no siempre de forma exacta (por restricciones de stock, por ejemplo).

Recomendación: Implementar alertas cuando la diferencia entre pedidos y ventas supere cierto umbral lo que es señal de posibles problemas logísticos o de inventario.

3. Inventario vs Unidades_Vendidas

Correlación baja o negativa ligera. Inventarios grandes no siempre implican mayores ventas, e incluso pueden reflejar sobre stock.

Recomendación: Optimizar el inventario a través de políticas de rotación y control de obsoletos.

4. Precio y Unidades_Vendidas

Suele haber una correlación negativa leve (-0.2 a -0.4), especialmente si hay sensibilidad al precio.

Recomendación: Realizar pruebas A/B de precios o análisis de elasticidad para fijar precios que maximicen ingresos.

5. Descuento y Unidades_Vendidas

Correlación positiva moderada si los descuentos influyen en el volumen de ventas (0.4 a 0.6).

Recomendación: Evaluar qué tipos de descuentos generan mayores retornos reales y evitar promociones poco efectivas o que canibalizan ingresos.

6. Precio_Competencia vs Unidades_Vendidas

Correlación negativa posible (-0.3 a -0.5). Si el precio de la competencia sube, podrías vender más, y viceversa.

Recomendación: Incorporar la variable de precios de la competencia en los dashboards y ajustar dinámicamente tus precios clave.

4.4 HISTOGRAMA CON KDE PARA CADA VARIABLE

Unidades_Vendidas

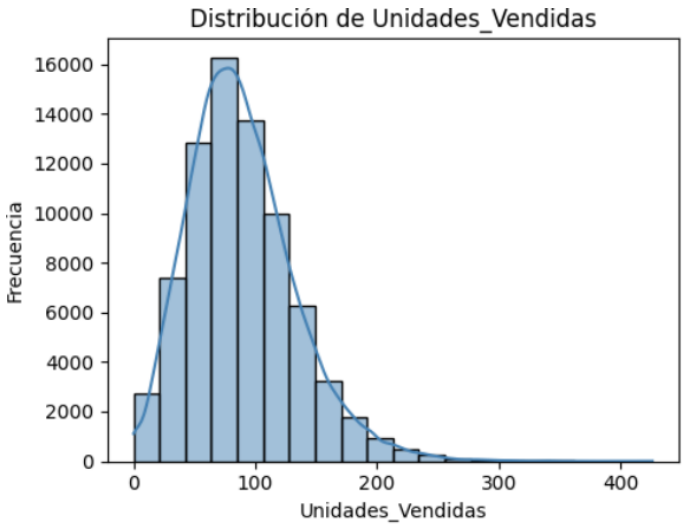


Gráfico 6 (Elaboración Propia).

Distribución sesgada a la derecha, muchos productos con pocas unidades vendidas, pocos productos con ventas muy altas.

Interpretación: Alta concentración de ventas bajas, productos de baja rotación. Colas largas pueden deberse a outliers o superventas estacionales.

Recomendación: Clasificar productos tipo ABC: A (altas ventas), B (medias), C (bajas). Enfocar promociones y espacio en tienda en productos tipo A.

Unidades_Pedidas

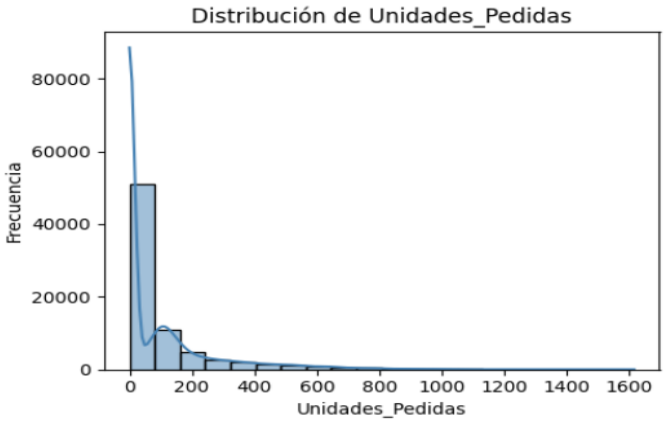


Gráfico 7 (Elaboración Propia).

Distribución similar a Unidades_Vendidas, aunque puede incluir más dispersión si hay roturas de stock.

Interpretación: Diferencia entre pedidos y ventas revela oportunidades perdidas, poca disponibilidad, mala logística.

Recomendación: Identificar productos con alta demanda no satisfecha. Mejorar la gestión de stock y previsión de demanda.

Inventario

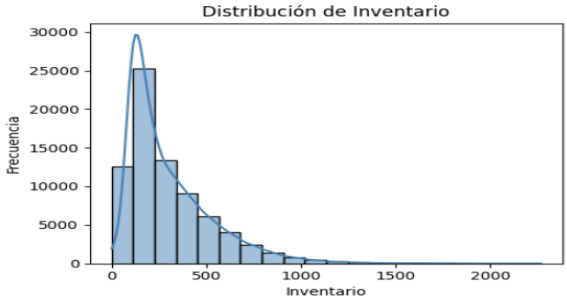


Gráfico 8 (Elaboración Propia).

Puede tener varios picos, reflejando ciclos de reposición o excesos.

Interpretación: Valores muy altos podrían indicar sobre stock o baja rotación. Valores muy bajos podrían limitar ventas.

Recomendación: Implementar control de inventario basado en rotación y demanda. Evitar capital inmovilizado en inventario obsoleto.

Precio

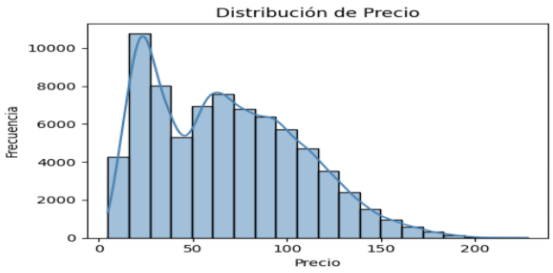


Gráfico 9 (Elaboración Propia).

Distribución multimodal si hay productos de gamas distintas (económica, media, alta).

Interpretación: Múltiples picos, segmentos de precio diferenciados. Dispersión muy alta puede confundir al consumidor.

Recomendación: Clarificar la propuesta de valor por segmento. Realizar análisis de elasticidad para ajustar precios.

Precio_Competencia

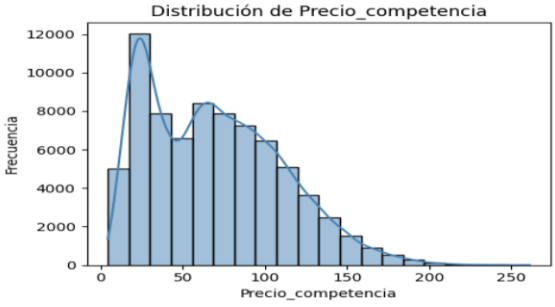


Gráfico 10 (Elaboración Propia).

Similar a Precio, aunque con diferentes modas dado a que el posicionamiento es distinto.

Interpretación: Precios están por encima de la media, valida la percepción de valor, lo justifica.

Recomendación: Comparar precios propios vs. competencia por categoría. Ajustar estrategia de precios dinámicamente.

Demanda

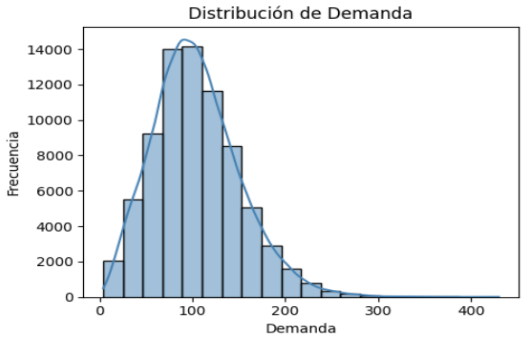


Gráfico 11 (Elaboración Propia).

Distribución sesgada, con muchos productos de baja demanda y pocos de alta.

Interpretación: Común en retail. Se pueden identificar oportunidades de crecimiento en productos con demanda potencial pero poca venta.

Recomendación: Invertir en marketing o reubicación de productos con alta demanda no cubierta. Utilizar modelos predictivos para alinear stock y demanda.

Descuento

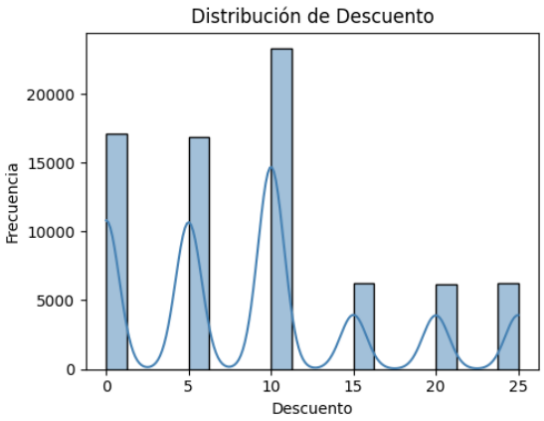


Gráfico 12 (Elaboración Propia).

Pico fuerte en 0 (productos sin descuento), y otros picos en valores promocionales comunes (10%, etc.).

Interpretación: La mayoría de los productos no tienen descuento en condiciones normales. Descuentos agresivos pueden generar caídas de margen si no se controlan.

Recomendación: Evaluar el impacto de cada nivel de descuento en las ventas. Aplicar descuentos solo a productos sensibles al precio.

4.5 GRAFICO DE BARRAS DE CATEGORIA MÁS VENDIDA

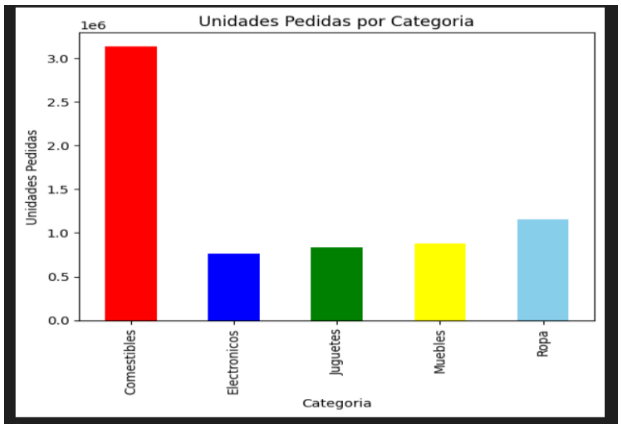


Gráfico 13 (Elaboración Propia).

Se puede observar que la categoría comestible es la que domina las ventas totales. El resto se reparte en proporciones menores o incluso marginales.

Interpretación: Las categorías con mayores ventas podrían representar productos esenciales o de alta rotación.
Mientras que las categorías con ventas bajas podrían tener:

- Bajo stock o visibilidad.
- Menor demanda del mercado.
- Precio o promoción inadecuados.

Recomendación: Para la categoría más vendida se debe asegurar stock, mantener o potenciar campañas de marketing. Para la segunda se debe evaluar potencial de crecimiento con promociones cruzadas. Pero para el resto de las categorías con ventas mas bajas se debe analizar si deben eliminarse, reposicionarse o promocionarse.

5. CONCLUSIÓN

El proceso de **recorte de outliers mediante IQR** ha sido efectivo.

Las **distribuciones se han centrado** alrededor de sus rangos típicos, sin distorsiones por valores extremos.

Esto mejorará notablemente el rendimiento e interpretación de:

- Modelos de regresión o predicción.
- Análisis de correlación y causalidad.
- Visualizaciones más limpias y comparables.

Los histogramas ayudaron a:

- Detectar sesgos, outliers y agrupamientos.
- Ajustar estrategias de precios, inventario y promoción.
- Priorizar productos según su desempeño y comportamiento de mercado.

6. BIBLIOGRAFÍA

Repositorio data CSV. KEAGGLE, *https://www.kaggle.com/datasets/atomicd/retail-store-inventory-and-demand-forecasting*

ORACLE, IQR (Rango Intercuartílico). *https://docs.oracle.com/cloud/help/es/pbcs_common/PFUSU/insights_metrics_IQR.htm#PFUSU-GUID-CF37CAEA-730B-4346-801E-64612719FF6B*