# Sales Data Analysis Report

**Name:** Nilesh Singh Yadav

**Roll Number:** 202401100400129

**Course:** Introduction to AI

**Date**: 10/03/2025

**Introduction:-** This report looks at sales trends over the year to understand how revenue patterns, product demand, and seasonal effects to help business optimize strategies.

## Key Findings:

### 1. Revanue Trends:-

- Sales peak in November and December due to holiday shopping.
- Revanue increases by 50% during the holiday season.
- A mid-year slump occurs from May to August.

### 2. Product Sales Patterns:-

- Laptops and smartphones generate the highest revenue.
- Accessories and Wearables have steady but lower sales.
- Discounts boost sales for accessories during promotions.

### 3. Seasonal Sales Patterns:-

- **Peak sales: -** Black Friday, Cyber Monday, and holidays.
- **Lower sales:-** Mid-year due to reduced consumer spending.

- Electronics lead in revenue all year.

## Business Implications :

- Increase marketing before shopping seasons.
- Stock up on popular products before peak sales.
- Use promotions to maintain sales during slow months.
- Engage costumers with loyalty programs and special offers.

## Conclusions:

Recognizing seasonal sales trends helps business plan inventory, improve marketing, and maximize revenue. Adapting strategies to high and low sales periods ensures steady business growth.

## CODE:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from datetime import datetime, timedelta

# Set random seed for reproducibility
np.random.seed(42)

# Generate date range for a full year
dates = pd.date_range(start='2024-01-01', periods=365, freq='D')
```

```
# Define product categories and their types
products = ['Laptop', 'Smartphone', 'Tablet', 'Headphones',
'Smartwatch']
categories = ['Electronics', 'Electronics', 'Electronics',
'Accessories', 'Wearables']

# Initialize an empty list to store sales data
sales_data = []

# Generate synthetic sales data
for date in dates:
    for i, product in enumerate(products):
        units_sold = np.random.randint(5, 50)  # Generate random demand
between 5 and 50 units
        price = np.random.randint(100, 1000)  # Generate random price
between 100 and 1000
        revenue = units_sold * price  # Calculate revenue
        sales_data.append([date, product, categories[i], units_sold,
revenue])

# Create a DataFrame from the generated sales data
df = pd.DataFrame(sales_data, columns=['Date', 'Product', 'Category',
'Units Sold', 'Revenue'])

# Introduce seasonality: Increase revenue by 50% for sales in November
and December
df.loc[df['Date'].dt.month.isin([11, 12]), 'Revenue'] *= 1.5

# Display first 10 rows of the dataset for verification
print("First 10 rows of the dataset:")
print(df.head(10))

# Adjust display settings for better clarity in Google Colab
pd.set_option("display.float_format", "{:.2f}".format)
pd.set_option("display.max_rows", 20)
pd.set_option("display.max_columns", None)

# Function to plot daily revenue trend
def plot_revenue_trend():
    plt.figure(figsize=(12, 5))
    df.groupby('Date')['Revenue'].sum().plot()
    plt.title('Daily Revenue Trend')
    plt.xlabel('Date')
    plt.ylabel('Total Revenue')
    plt.grid()
    plt.show()

# Function to visualize total units sold per product
```

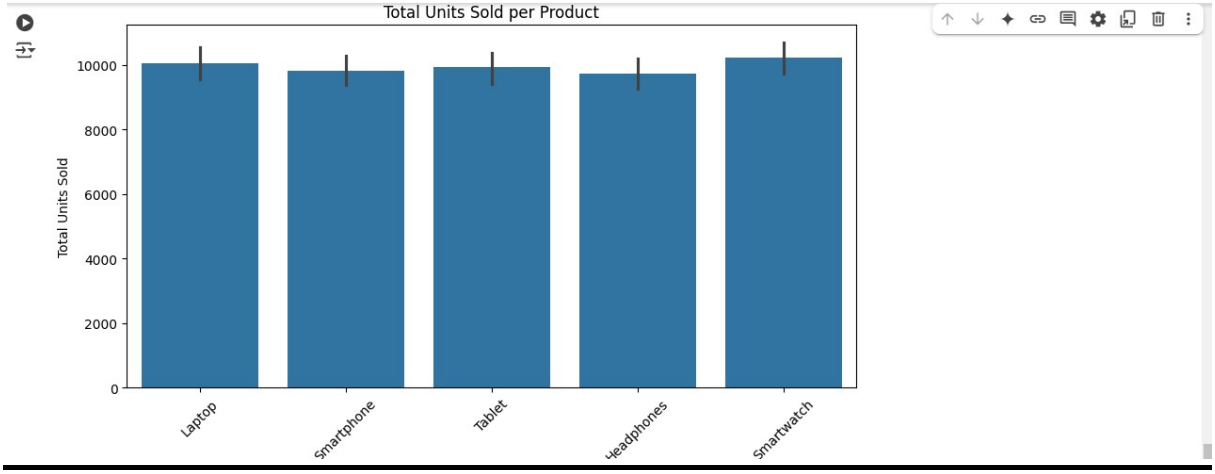
```
def plot_product_demand():
    plt.figure(figsize=(10, 5))
    sns.barplot(x=df['Product'], y=df['Units Sold'], estimator=sum)
    plt.title('Total Units Sold per Product')
    plt.ylabel('Total Units Sold')
    plt.xticks(rotation=45)
    plt.show()

# Function to visualize seasonal revenue trends
def plot_seasonal_trends():
    df['Month'] = df['Date'].dt.month  # Extract month from date
    plt.figure(figsize=(10, 5))
    sns.lineplot(x='Month', y='Revenue', data=df, estimator=sum,
marker='o')
    plt.title('Seasonal Revenue Trends')
    plt.xlabel('Month')
    plt.ylabel('Total Revenue')
    plt.xticks(range(1, 13))  # Set x-axis labels from 1 to 12 (months)
    plt.show()

# Call the functions to generate visualizations
plot_revenue_trend()
plot_product_demand()
plot_seasonal_trends()

# Save the dataset as a CSV file
df.to_csv('sales_data.csv', index=False)
print("Dataset saved as sales_data.csv")
```

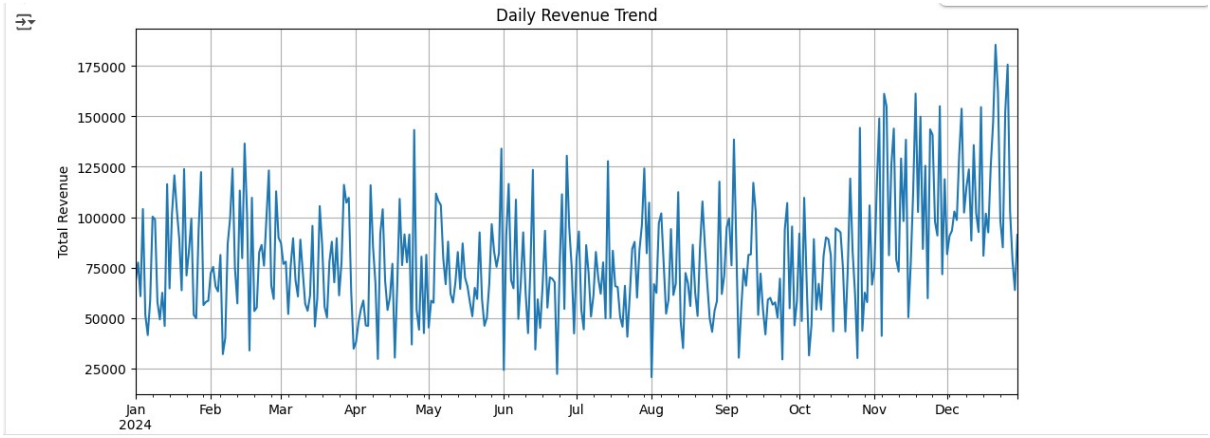
# **Credits**:

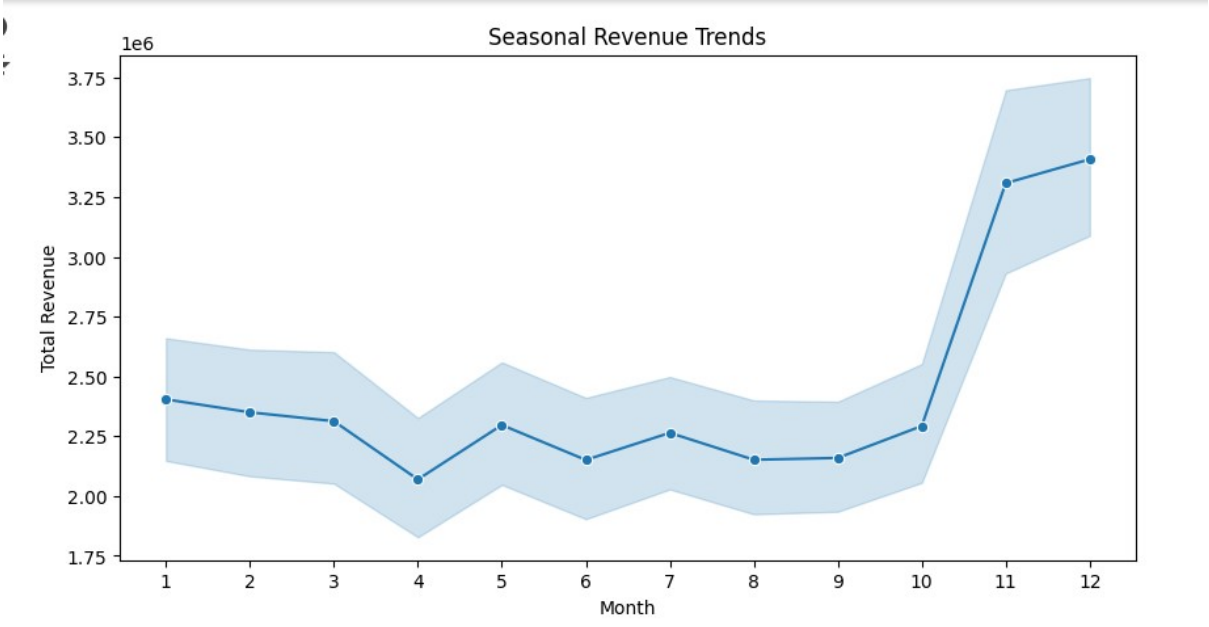
Dataset:- Generated from Chatbot.

Tools Used:- Python (Pandas, Matplotlib, Seaborn), Jupyter Notebook, Google colab, Chatbot.

# Output:


Total Units Sold per Product
Total Units Sold
0
2000
4000
6000
8000
10000
Laptop
Smartphone
Tablet
Headphones
Smartwatch


Daily Revenue Trend
Total Revenue
25000
50000
75000
100000
125000
150000
175000
Jan
2024
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec


Seasonal Revenue Trends
1e6
Total Revenue
1.75
2.00
2.25
2.50
2.75
3.00
3.25
3.50
3.75
1
2
3
4
5
6
7
8
9
10
11
12
Month