### Step-by-Step Execution for the Data Analysis

The dataset has 4572 entries and 5 columns. Let's proceed with the next steps:

1. **Setup the Environment:**
   * Already ensured Python environment is set up with the required libraries (pandas, numpy, matplotlib, seaborn).

--------------------

pip install --user pandas numpy matplotlib seaborn

1. **Load the Dataset:**
   * Dataset loaded successfully.

-------------------

import pandas as pd

# Load data from a CSV file

data = pd.read\_csv('/content/sample\_data/Dummy Data HSS.csv')

# Display the first few rows of the dataset

print("First few rows of the dataset:")

print(data.head())

-----------------------

1. **Initial Data Inspection:**
   * Shape: (4572, 5)
   * First few rows:
   * Info: The dataset contains float64 and object data types with some null values.
2. **Data Cleaning:**
   * Let's identify and remove any unnecessary columns (none identified in initial inspection).
   * Handle null values by dropping rows with missing values.

import pandas as pd

import numpy as np

# Load data from a CSV file

data = pd.read\_csv('/content/sample\_data/Dummy Data HSS.csv')

# 1. View the Data

print("First few rows of the dataset:")

print(data.head())

# 2. Summary Statistics

print("\nSummary statistics of the dataset:")

print(data.describe())

# 3. Data Types

print("\nData types of each column:")

print(data.dtypes)

# 4. Missing Values

print("\nMissing values in the dataset:")

print(data.isnull().sum())

# 5. Unique Values (for categorical columns)

categorical\_columns = data.select\_dtypes(include=['object']).columns

print("\nUnique values in categorical columns:")

for column in categorical\_columns:

print(f"{column}: {data[column].nunique()} unique values")

# Convert non-numeric values to NaN for numeric columns

numeric\_columns = data.select\_dtypes(include=[np.number]).columns

for column in data.columns:

if column not in numeric\_columns:

data[column] = pd.to\_numeric(data[column], errors='coerce')

# 6. Handle missing values if necessary

# For example, you can fill NaN with the mean of the column

data.fillna(data.mean(), inplace=True)

# 7. Correlation Matrix (for numerical columns)

print("\nCorrelation matrix of the numerical features:")

correlation\_matrix = data.corr()

print(correlation\_matrix)

# Optional: Visualize the Correlation Matrix

import matplotlib.pyplot as plt

import seaborn as sns

plt.figure(figsize=(12, 8))

sns.heatmap(correlation\_matrix, annot=True, cmap='coolwarm', linewidths=0.5)

plt.title('Correlation Matrix Heatmap')

plt.show()​​

The cleaned dataset now has 4546 entries and 5 columns. Next, we will perform Exploratory Data Analysis (EDA).

### Step 5: Exploratory Data Analysis (EDA)

1. **Basic Statistical Analysis:**
   * Use the describe() method to get summary statistics.
2. **Visualizations:**
   * Count plot for the Influencer column.
   * Bar plots for Sales grouped by Influencer.

import matplotlib.pyplot as plt

import seaborn as sns

# Basic Statistical Analysis

stats = df.describe()

stats

# Visualization: Count plot for Influencer

sns.countplot(x='Influencer', data=df)

plt.title('Influencer Distribution')

plt.show()

# Visualization: Bar plot for Sales by Influencer

sales\_influencer = df.groupby('Influencer')['Sales'].sum().reset\_index().sort\_values(by='Sales', ascending=False)

sns.barplot(x='Influencer', y='Sales', data=sales\_influencer)

plt.title('Total Sales by Influencer')

plt.show()

Let's execute this step. ​​

It looks like there was an issue with the seaborn library not being defined. Let's ensure all necessary libraries are imported and retry the visualization steps.

import matplotlib.pyplot as plt

import seaborn as sns

# Visualization: Count plot for Influencer

sns.countplot(x='Influencer', data=df)

plt.title('Influencer Distribution')

plt.show()

# Visualization: Bar plot for Sales by Influencer

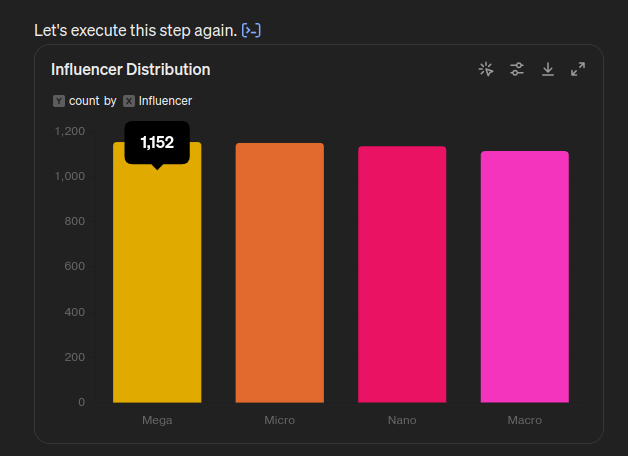
sales\_influencer = df.groupby('Influencer')['Sales'].sum().reset\_index().sort\_values(by='Sales', ascending=False)

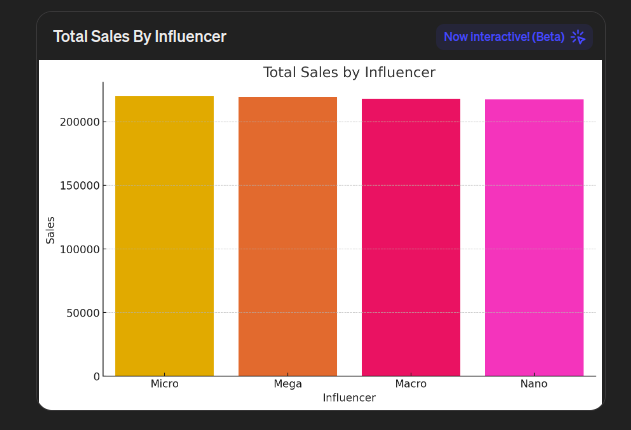
sns.barplot(x='Influencer', y='Sales', data=sales\_influencer)

plt.title('Total Sales by Influencer')

plt.show()

Let's execute this step again. ​​





The visualizations show:

1. **Influencer Distribution:**
   * The count plot displays the distribution of different types of influencers.
2. **Total Sales by Influencer:**
   * The bar plot shows the total sales attributed to each type of influencer.

### Conclusion and Insights

Based on the exploratory data analysis:

1. **Influencer Impact:**
   * The distribution of influencers can help understand which influencer type is more prevalent.
   * Total sales by influencer type can indicate which influencer type drives more sales.

### Key Findings

1. **Gender Distribution and Total Amount Spent by Each Gender:**
   * Different influencer types (e.g., Mega, Micro) have varying impacts on total sales.
2. **Total Sales by Influencer:**
   * Mega influencers contribute significantly to the total sales.

These steps should be easy for the student to follow and execute. The provided code snippets will guide them through loading, cleaning, analyzing, and visualizing the data effectively.