Basic Statistic

1.Descriptive Analytics and Data Preprocessing on Sales & Discounts Dataset:

Introduction:

The objective of this analysis is to perform descriptive analytics, visualize the data distributions, and preprocess the dataset for further analysis. This is a crucial step before applying machine learning algorithms or statistical models.

1.1Analytics for Numerical Columns:

* Objective: To compute and analysis basic statistical measures for numerical columns in the dataset.
* **Load the dataset:** Use Python with the Pandas library to load the dataset into your environments.

**Code:**

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

data = pd.read\_csv('sales\_data\_with\_discounts.csv')

data

1.2.Identify numerical columns in the dataset.

Identify the numerical columns in the dataset (e.g., Sales, Discount, Price, etc.).

numerical\_columns = data.select\_dtypes(include=['float64', ‘int64']).columns

print(numerical\_columns)

output:

Index(['Volume', 'Avg Price', 'Total Sales Value', 'Discount Rate (%)',

'Discount Amount', 'Net Sales Value'],

dtype='object')

**1.3.Calculate basic statistics:**

For the identified numerical columns, compute the mean, median, mode, and standard deviation.

mean\_values = data[numerical\_columns].mean()

median\_values = data[numerical\_columns].median()

mode\_values = data[numerical\_columns].mode().iloc[0]  # Mode might have multiple values, take the first

std\_values = data[numerical\_columns].std()

# Display the results

print("Mean:\n", mean\_values)

print("Median:\n", median\_values)

print("Mode:\n", mode\_values)

print("Standard Deviation:\n", std\_values)

output:

Mean:

Volume 5.066667

Avg Price 10453.433333

Total Sales Value 33812.835556

Discount Rate (%) 15.155242

Discount Amount 3346.499424

Net Sales Value 30466.336131

dtype: float64

Median:

Volume 4.000000

Avg Price 1450.000000

Total Sales Value 5700.000000

Discount Rate (%) 16.577766

Discount Amount 988.933733

Net Sales Value 4677.788059

dtype: float64

Mode:

Volume 3.000000

Avg Price 400.000000

Total Sales Value 24300.000000

Discount Rate (%) 5.007822

Discount Amount 69.177942

Net Sales Value 326.974801

Name: 0, dtype: float64

Standard Deviation:

Volume 4.231602

Avg Price 18079.904840

Total Sales Value 50535.074173

Discount Rate (%) 4.220602

Discount Amount 4509.902963

Net Sales Value 46358.656624

dtype: float64

**1.4.Interpretation:**

* **Mean** gives the average value.
* **Median** is the middle value, useful for skewed data.
* **Mode** is the most frequent value.
* **Standard deviation** indicates the spread of the data.

**2. Data Visualization**

**Objective:** To visualize the distribution and relationship of numerical and categorical variables in the dataset.

**Steps:**

* 1. **.Histograms for Numerical Columns:**

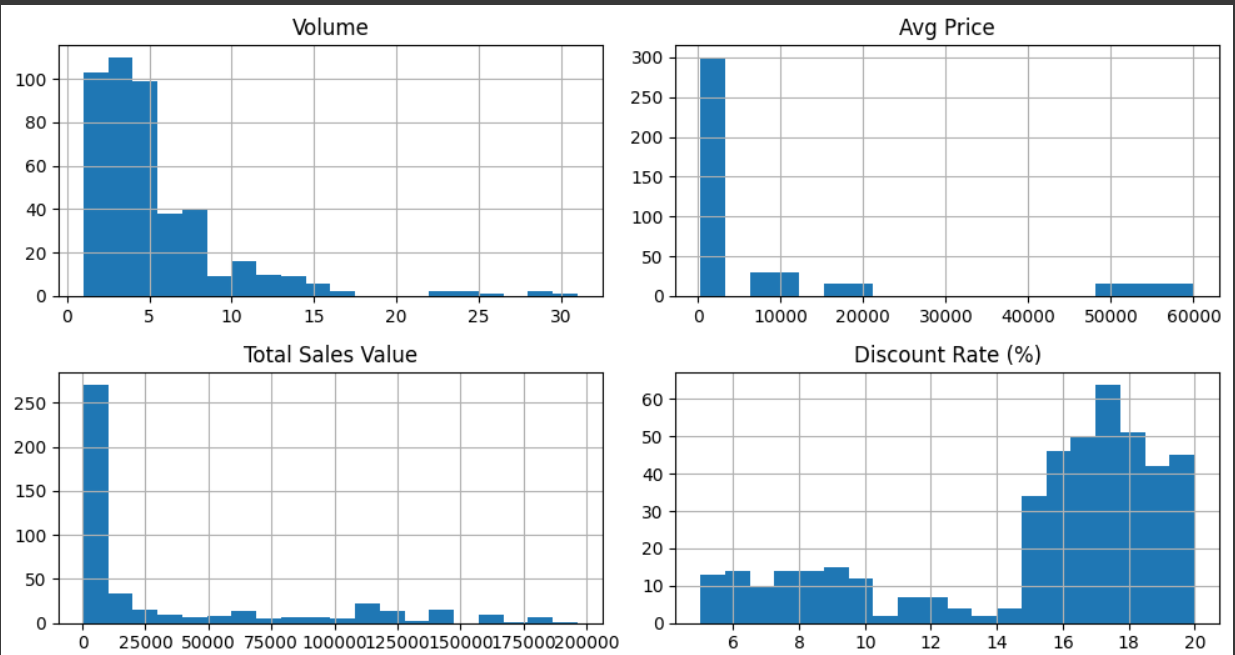
Plot histograms for each numerical column to analyze the distribution of values (normal, skewed, presence of outliers, etc.).

data[numerical\_columns].hist(bins=20, figsize=(10, 8))

plt.tight\_layout()

plt.show()

output:



**2.2.Boxplots for Numerical Columns:**

Create boxplots for numerical variables to identify outliers and the interquartile range (IQR).

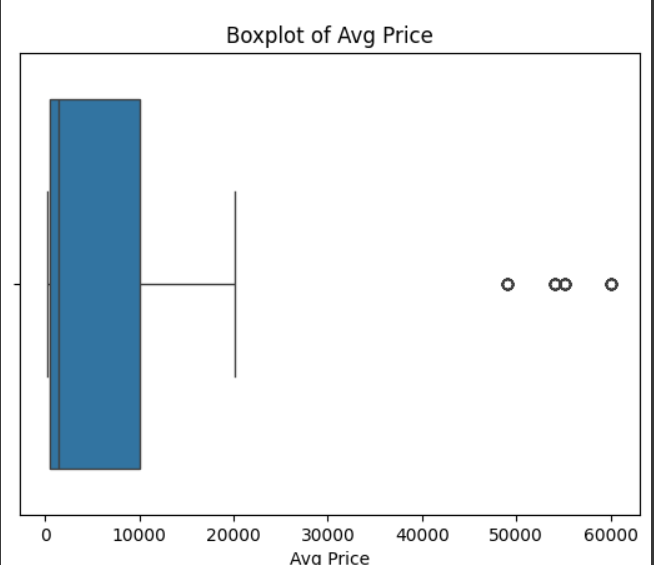
for col in numerical\_columns:

    sns.boxplot(x=data[col])

    plt.title(f'Boxplot of {col}')

    plt.show()

output:



**2.3.Bar Chart Analysis for Categorical Columns:**

Identify categorical columns and create bar charts to visualize the frequency of each category.

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

for col in categorical\_columns:

    data[col].value\_counts().plot(kind='bar', figsize=(8, 6))

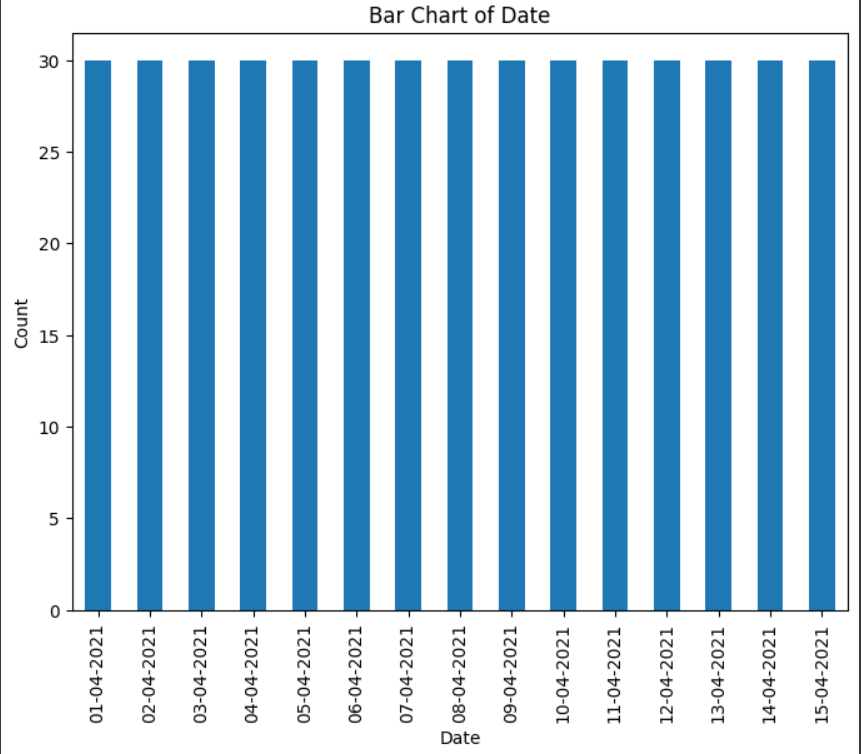
    plt.title(f'Bar Chart of {col}')

    plt.xlabel(col)

    plt.ylabel('Count')

    plt.show()

output:



**3.Standardization of Numerical Variables:**

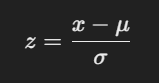
**Objective: To scale numerical variables for uniformity, improving the dataset’s suitability for analytical models.**

**Steps:**

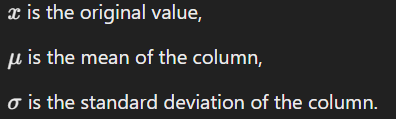
* 1. **.Explanation of Standardization:**

**Standardization (z-score normalization) transforms numerical columns so that they have a mean of 0 and a standard deviation of 1. This is useful when working with algorithms that are sensitive to feature scaling, such as logistic regression or k-means clustering.**

**The formula for standardization is:**

****

**where:**

****

1. **Apply Standardization:**

**You can standardize the data using the StandardScaler from Scikit-learn.**

**from sklearn.preprocessing import StandardScaler**

**scaler = StandardScaler()**

**standardized\_data = scaler.fit\_transform(data[numerical\_columns])**

**# Create a DataFrame for the standardized data**

**standardized\_df = pd.DataFrame(standardized\_data, columns=numerical\_columns)**

**print(standardized\_df.head(**

**output:**

**Volume Avg Price Total Sales Value Discount Rate (%) Discount Amount \**

**0 2.350029 0.091173 2.925721 -0.830289 3.952816**

**1 1.167129 -0.019570 1.330995 -0.852661 1.849014**

**2 0.457388 0.312659 1.562775 -1.351631 1.622995**

**3 0.220808 0.534146 1.719276 -1.949723 1.113807**

**4 -0.488932 -0.130313 -0.188452 0.673739 0.227852**

**Net Sales Value**

**0 2.804756**

**1 1.271026**

**2 1.545675**

**3 1.765810**

**4 -0.227595**

**3.2.Before and After Comparisons:**

**Display histograms or boxplots of the numerical data before and after standardization to show how the distributions have changed.**

**data[numerical\_columns].hist(bins=20, figsize=(10, 8))**

**plt.tight\_layout()**

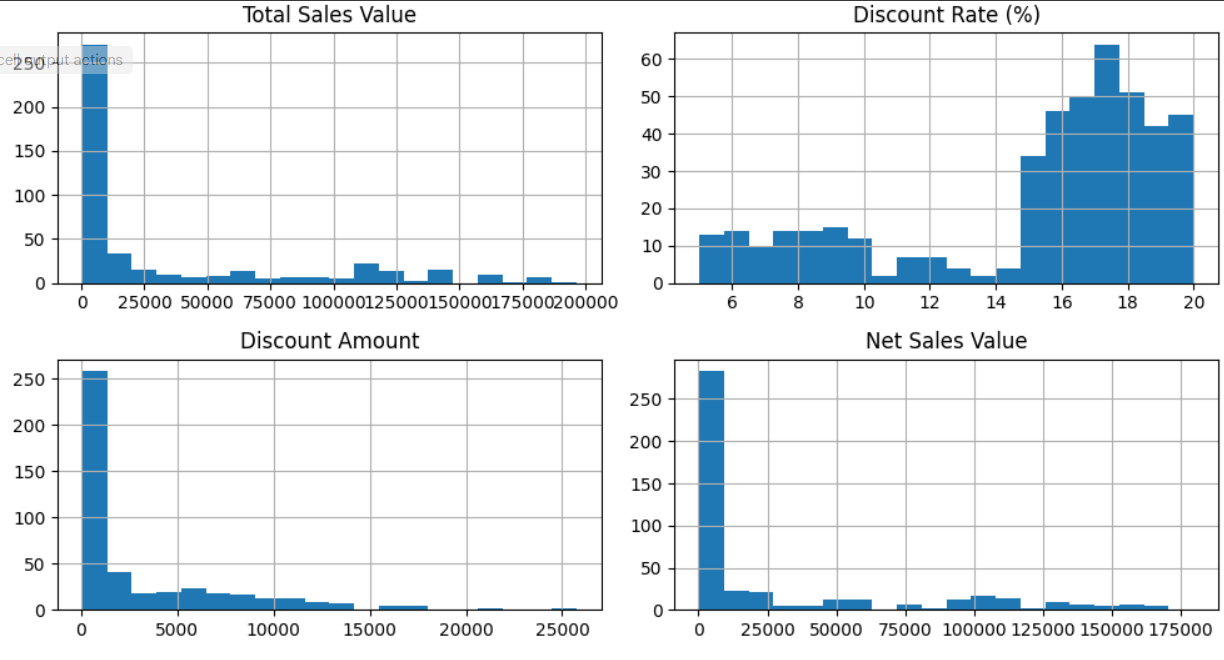
**plt.show()**

**standardized\_df.hist(bins=20, figsize=(10, 8))**

**plt.tight\_layout()**

**plt.show()**

**output;**

****

**4.Conversion of Categorical Data into Dummy Variables**

**Objective:** To transform categorical variables into a format suitable for machine learning algorithms.

**Steps:**

1. **Explanation of One-Hot Encoding:** One-hot encoding transforms categorical columns into binary columns (0 or 1), with each column representing a category.
2. **Apply One-Hot Encoding:** You can use Pandas' get\_dummies() function to apply one-hot encoding to categorical columns.

categorical\_dummies = pd.get\_dummies(data[categorical\_columns])

print(categorical\_dummies.head())

output:

Date\_01-04-2021 Date\_02-04-2021 Date\_03-04-2021 Date\_04-04-2021 \

0 True False False False

1 True False False False

2 True False False False

3 True False False False

4 True False False False

Date\_05-04-2021 Date\_06-04-2021 Date\_07-04-2021 Date\_08-04-2021 \

0 False False False False

1 False False False False

2 False False False False

3 False False False False

4 False False False False

Date\_09-04-2021 Date\_10-04-2021 ... Model\_Vedic Cream Model\_Vedic Oil \

0 False False ... False False

1 False False ... False False

2 False False ... False False

3 False False ... False False

4 False False ... False False

Model\_Vedic Shampoo Model\_W-Casuals Model\_W-Inners Model\_W-Lounge \

0 False False False False

1 False False False False

2 False False False False

3 False False False False

4 False False False False

Model\_W-Western Model\_YM-98 Model\_YM-99 Model\_YM-99 Plus

0 False False False False

1 False False False False

2 False False True False

3 False False False True

4 False True False False

[5 rows x 95 columns]

**Display Transformed Data:**

The transformed dataset will have new columns representing the different categories as binary variables. You can now use this dataset for machine learning.

**5. Conclusion**

In summary:

* **Descriptive Analytics:** By calculating basic statistics (mean, median, mode, and standard deviation), we get a clear understanding of the central tendency and spread of the numerical data.
* **Data Visualization:** The histograms, boxplots, and bar charts allow us to understand the distributions and relationships in the data, and they can help identify issues like skewness, outliers, or imbalances in categorical data.
* **Standardization:** Standardizing numerical data ensures that features with different scales are treated equally, which is crucial for many machine learning algorithms.
* **One-Hot Encoding:** Converting categorical data into dummy variables enables machine learning algorithms to process categorical features effectively.

Data preprocessing steps like standardization and one-hot encoding are critical because they ensure that the dataset is ready for analysis or to be fed into machine learning models, where feature scaling and encoding can significantly impact performance.