**Assignment: Descriptive Analytics and Data Preprocessing on Sales & Discounts Dataset**

**Introduction**

**Objective:**  
The objective of this assignment is to perform descriptive analytics to understand the dataset, visualize important patterns and distributions, and apply essential preprocessing techniques to prepare the dataset for further analytical or machine learning tas

**Task 1: Descriptive Analytics for Numerical Columns**

**Steps:**

1. **Load the Dataset**

python

import pandas as pd

df = pd.read\_csv('sales\_discounts.csv') # Replace with actual file path

1. **Identify Numerical Columns**

Python

numerical\_cols = df.select\_dtypes(include=['float64', 'int64']).columns

df[numerical\_cols].head()

1. **Compute Descriptive Statistics**

Python

desc\_stats = df[numerical\_cols].describe().T

desc\_stats['mode'] = df[numerical\_cols].mode().iloc[0]

desc\_stats[['mean', '50%', 'mode', 'std']] # 50% = median

**Interpretation:**

* **Mean**: Indicates the average sales or discount.
* **Median**: Useful for understanding the central value while minimizing the impact of outliers.
* **Mode**: Highlights the most frequent numerical values.
* **Standard Deviation**: Reflects the variability; a higher std shows more spread in the data.

**Task 2: Data Visualization**

**2.1 Histograms**

**Goal:** Visualize the distribution of numerical features.

python

import matplotlib.pyplot as plt

import seaborn as sns

for col in numerical\_cols

plt.figure(figsize=(6, 4))

sns.histplot(df[col], kde=True)

plt.title(f'Histogram of {col}')

plt.show()

**2.2 Boxplots**

**Goal:** Detect outliers and visualize IQR.

python

for col in numerical\_cols:

plt.figure(figsize=(6, 4))

sns.boxplot(x=df[col])

plt.title(f'Boxplot of {col}')

plt.show()

**2.3 Bar Charts for Categorical Columns**

**Goal:** Understand distribution of categorical variables.

python

categorical\_cols = df.select\_dtypes(include=['object']).columns

for col in categorical\_cols:

plt.figure(figsize=(6, 4))

df[col].value\_counts().plot(kind='bar')

plt.title(f'Bar Chart of {col}')

plt.xlabel(col)

plt.ylabel('Count')

plt.show()

**Task 3: Standardization of Numerical Variables**

**Concept:**

Standardization transforms data to have a **mean of 0** and **standard deviation of 1** using the formula:

z=x−μσz = \frac{x - \mu}{\sigma}z=σx−μ​

Where:

* xxx: individual value
* μ\muμ: mean
* σ\sigmaσ: standard deviation

**Implementation:**

python

from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

df\_scaled = df.copy()

df\_scaled[numerical\_cols] = scaler.fit\_transform(df[numerical\_cols])

# Visualize before vs after

df[[numerical\_cols[0]]].hist()

df\_scaled[[numerical\_cols[0]]].hist()

**Before vs After:**

* Raw data may have inconsistent scales.
* Standardized data has uniform scale, improving model fairness and convergence.

**Task 4: Conversion of Categorical Data into Dummy Variables**

**Why One-Hot Encoding?**

Machine learning models require numerical inputs. One-hot encoding prevents incorrect assumptions of order in categorical variables.

**Implementation:**

python

df\_encoded = pd.get\_dummies(df, columns=categorical\_cols, drop\_first=True)

df\_encoded.head()

**Sample Output Table:**

| **Sales** | **Discount** | **Category\_Furniture** | **Category\_Technology** |
| --- | --- | --- | --- |
| 200 | 0.1 | 1 | 0 |

**Conclusion**

**Key Findings:**

* Descriptive analytics revealed central tendencies and data variability.
* Histograms and boxplots identified skewed data and outliers.
* Bar charts provided insights into category distributions.

**Importance of Preprocessing:**

* **Standardization**: Essential for models that rely on scale (e.g., KNN, regression).
* **One-hot Encoding**: Makes categorical data compatible with modeling, improving interpretability and accuracy.