Day-20

**Cleaning and Preparing an E-Commerce Customer Reviews Dataset**

**Objective:**

To clean and prepare a dataset containing customer reviews and ratings of e-commerce products.

**Instructions:**

1. **Load the dataset** containing customer reviews, ratings, and feedback.
2. **Handle missing values**:
   * Identify missing values in Review\_Text, Rating, and Customer\_Age.
   * Impute missing numerical values using appropriate techniques.
   * Use NLP-based techniques to handle missing textual data.
3. **Detect and remove duplicates**:
   * Use duplicated() to find repeated reviews.
   * Remove or merge duplicate records.
4. **Handle inconsistent data**:
   * Standardize Rating values (ensure they range between 1-5).
   * Correct spelling inconsistencies in Product\_Category.
5. **Identify and handle outliers**:
   * Use boxplots to find anomalies in Product\_Price and Rating.
   * Apply transformation techniques if necessary.
6. **Prepare cleaned data for analysis**:
   * Convert categorical data into numerical format where required.
   * Save the final cleaned dataset as a CSV file.

Program:

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

from sklearn.impute import SimpleImputer

from sklearn.preprocessing import LabelEncoder

from sklearn.preprocessing import StandardScaler

1. Load the Dataset

df = pd.read\_csv('ecommerce\_reviews.csv') # Replace with your actual file path

df.head()

2. Identify Missing Data

missing\_data = df.isna().sum()

missing\_percentage = (missing\_data / len(df)) \* 100

print("Missing Data Count:\n", missing\_data)

print("\nMissing Data Percentage:\n", missing\_percentage)

# Get general info about the dataset

df.info()

3. Handle Missing Values

# Impute Numerical Columns (e.g., Rating, Customer\_Age)

imputer = SimpleImputer(strategy='mean') # Mean imputation for numerical columns

df['Rating'] = imputer.fit\_transform(df[['Rating']])

df['Customer\_Age'] = imputer.fit\_transform(df[['Customer\_Age']])

# Use NLP-based Techniques for Missing Review\_Text

# We could use a placeholder to fill missing reviews

df['Review\_Text'] = df['Review\_Text'].fillna('No Review')

4. Detect and Remove Duplicates

# Check for duplicated reviews

duplicates = df[df.duplicated(subset=['Review\_Text', 'Rating'], keep=False)]

print("\nDuplicated Reviews:\n", duplicates)

# Remove duplicate reviews

df = df.drop\_duplicates(subset=['Review\_Text', 'Rating'], keep='first')

5. Handle Inconsistent Data

# Standardize Rating values (ensure they range between 1-5)

df['Rating'] = df['Rating'].clip(lower=1, upper=5)

# Correct spelling inconsistencies in Product\_Category

df['Product\_Category'] = df['Product\_Category'].str.lower() # Convert to lowercase

df['Product\_Category'] = df['Product\_Category'].replace({'electronics': 'electronic', 'fashion': 'clothing'})

6. Identify and Handle Outliers

# Use Boxplots to Identify Outliers in Product\_Price and Rating

plt.figure(figsize=(12, 6))

# Boxplot for Product\_Price

sns.boxplot(x=df['Product\_Price'])

plt.title('Boxplot for Product\_Price')

plt.show()

# Boxplot for Rating

sns.boxplot(x=df['Rating'])

plt.title('Boxplot for Rating')

plt.show()

# Handle Outliers (e.g., Cap values outside of 1.5\*IQR)

Q1\_price = df['Product\_Price'].quantile(0.25)

Q3\_price = df['Product\_Price'].quantile(0.75)

IQR\_price = Q3\_price - Q1\_price

lower\_bound\_price = Q1\_price - 1.5 \* IQR\_price

upper\_bound\_price = Q3\_price + 1.5 \* IQR\_price

df['Product\_Price'] = df['Product\_Price'].clip(lower=lower\_bound\_price, upper=upper\_bound\_price)

Q1\_rating = df['Rating'].quantile(0.25)

Q3\_rating = df['Rating'].quantile(0.75)

IQR\_rating = Q3\_rating - Q1\_rating

lower\_bound\_rating = Q1\_rating - 1.5 \* IQR\_rating

upper\_bound\_rating = Q3\_rating + 1.5 \* IQR\_rating

df['Rating'] = df['Rating'].clip(lower=lower\_bound\_rating, upper=upper\_bound\_rating)

7. Convert Categorical Data into Numerical Format

# 7.1 Convert 'Product\_Category' using Label Encoding

label\_encoder = LabelEncoder()

df['Product\_Category'] = label\_encoder.fit\_transform(df['Product\_Category'])

Output:

Missing Data Count:

Review\_Text 0

Rating 50

Customer\_Age 100

Missing Data Percentage:

Review\_Text 0.0%

Rating 5.0%

Customer\_Age 10.0%

# 7.2 Standardize Numerical Data (e.g., Product\_Price) using Standard Scaling

scaler = StandardScaler()

df['Product\_Price'] = scaler.fit\_transform(df[['Product\_Price']])

# 8. Save the Cleaned Dataset

df.to\_csv('cleaned\_ecommerce\_reviews.csv', index=False)

# Print the final cleaned dataset

print("\nCleaned Dataset Saved as 'cleaned\_ecommerce\_reviews.csv'")