Batch: D-2 **Roll No.:** 16010122151

Title: Implement data pre-processing using python on real world dataset

Course Outcome:

CO1 Understand basic concepts of data analytics to solve real-world problems

Books/ Journals/ Websites referred:

Geeksforgeeks.com

Resources used:

Google colab for writing python scripts

Theory (About Data PreprData preprocessing is essential for preparing raw data for analysis and modeling. It involves:

1. Data Cleaning:

- Handling Missing Values: Using techniques like deletion or imputation to address gaps in data.

- Removing Duplicates: Ensuring no repeated data points.

- Outlier Detection: Identifying and managing unusually extreme values.

2. Data Transformation:

-Normalization/Standardization: Scaling data to ensure consistency, especially for algorithms sensitive to feature magnitudes.

- Encoding Categorical Data: Converting categorical variables into numerical formats, like one-hot encoding.

- Feature Engineering: Creating new features or modifying existing ones to improve model performance.

3. Data Reduction:

- Dimensionality Reduction: Reducing the number of features using methods like PCA to eliminate redundant data.

- Feature Selection: Selecting the most relevant features to enhance model efficiency and accuracy.

Preprocessing ensures that the data is clean, consistent, and ready for effective analysis or modeling.ocessing):

Program:

```
import pandas as pd

import numpy as np

# Sample data

data = {

    'name': ['Alice', 'Bob', 'Charlie', 'Dave', 'Eve'],

    'age': [25, np.nan, 30, 22, 35],

    'gender': ['F', 'M', 'M', 'M', 'F'],

    'income': [50000, 60000, 75000, np.nan, 80000]

}

df = pd.DataFrame(data)
```

```
# Display the original data
print("Original DataFrame:")
print(df)
# User-defined function for discretization
def discretize_age(age):
    if age < 30:
        return 'Young'
    elif age >= 30 and age < 40:
        return 'Middle-aged'
    else:
        return 'Old'


# Handling missing values (NaN)
# Fill missing values in 'age' with the mean age
mean_age = df['age'].mean()
df['age'].fillna(mean_age, inplace=True)
# Apply discretization function to 'age' column
df['age_category'] = df['age'].apply(discretize_age)
# Drop rows with missing values in any column
df.dropna(inplace=True)
# Convert categorical variables (gender) to numerical
df['gender'] = df['gender'].map({'F': 0, 'M': 1})
# Data normalization Min -Max
# Normalize 'income' column to range [0, 1]
min_income = df['income'].min()
max_income = df['income'].max()
```

```
df['income_normalized'] = (df['income'] - min_income) / (max_income - min_income)
```

```
# Display cleaned, preprocessed, and discretized data
print("\nCleaned, Preprocessed, and Discretized DataFrame:")
print(df)
```

—--

Task: Download the real time data set and implement data preprocessing techniques on the real time data set

Source of the dataset (URL):

Platform used by the student:

Following points should be written by students

Different steps in Data Preprocessing:

- **Finding missing, null values**
- **Replacing missing, null values with statistical parameters**
- **Encoding categorical data if needed (Write user defined function)**
- **Normalization (Write user defined function)**
- **Discretization (Write user defined function)**

Working (Paste the code and Output for each Data Preprocessing task):

Students need to write comments wherever needed

```
df['Age_Group'] = pd.cut(df['Age'], bins=[0, 0.3, 0.6, 0.9, 1.0], labels=['Young', 'Adult', 'Middle-Aged', 'Senior'])
df.head()
```

_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Category	Orders	Amount	Age_Group
anskriti	P00125942	F	26-35	0.2000	0	Maharashtra	Western	Healthcare	Auto	0.000000	1.000000	Young
Kartik	P00110942	F	26-35	0.2875	1	Andhra Pradesh	Southern	Govt	Auto	0.666667	0.999243	Young
Bindu	P00118542	F	26-35	0.2875	1	Uttar Pradesh	Central	Automobile	Auto	0.666667	0.998822	Young
Sudevi	P00237842	M	0-17	0.0500	0	Karnataka	Southern	Construction	Auto	0.333333	0.998317	Young
Joni	P00057942	M	26-35	0.2000	1	Gujarat	Western	Food Processing	Auto	0.333333	0.996844	Young

```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
df[['Age', 'Orders', 'Amount']] = scaler.fit_transform(df[['Age', 'Orders', 'Amount']])
df.describe()
```

	User_ID	Age	Marital_Status	Orders	Amount
count	1.125100e+04	11251.000000	11251.000000	11251.000000	11251.000000
mean	1.003004e+06	0.292765	0.420318	0.496430	0.389901
std	1.716125e+03	0.159427	0.493632	0.371682	0.219642
min	1.000001e+06	0.000000	0.000000	0.000000	0.000000
25%	1.001492e+06	0.187500	0.000000	0.166667	0.221154
50%	1.003065e+06	0.262500	0.000000	0.333333	0.333361
75%	1.004430e+06	0.387500	1.000000	0.666667	0.525290
max	1.006040e+06	1.000000	1.000000	1.000000	1.000000

```
[14] pd.isnull(df).sum()
```

```
User_ID             0
Cust_name           0
Product_ID          0
Gender              0
Age Group           0
Age                 0
Marital_Status      0
State               0
Zone                0
Occupation          0
Product_Category    0
Orders              0
Amount              0
dtype: int64
```

```
df[['Age', 'Orders', 'Amount']].describe()
```

	Age	Orders	Amount
count	11251.000000	11251.000000	11251.000000
mean	35.421207	2.489290	9453.610858
std	12.754122	1.115047	5219.569870
min	12.000000	1.000000	188.000000
25%	27.000000	1.500000	5443.500000
50%	33.000000	2.000000	8110.000000

```
pd.isnull(df).sum()

User_ID              0
Cust_name            0
Product_ID           0
Gender               0
Age Group            0
Age                  0
Marital_Status       0
State                0
Zone                 0
Occupation           0
Product_Category     0
Orders               0
Amount              12
dtype: int64

[12] mean_age = df['Amount'].mean()

[13] df['Amount'].fillna(mean_age, inplace=True)
```

```
[6] pd.isnull(df).sum()

User_ID                 0
Cust_name               0
Product_ID              0
Gender                  0
Age Group               0
Age                     0
Marital_Status          0
State                   0
Zone                    0
Occupation              0
Product_Category        0
Orders                  0
Amount                 12
Status              11251
unnamed1            11251
dtype: int64

[8] df.drop(['Status', 'unnamed1'], axis=1, inplace=True)
```

```
[4] df.describe()
```

	User_ID	Age	Marital_Status	Orders	Amount	Status	unnamed1
count	1.125100e+04	11251.000000	11251.000000	11251.000000	11239.000000	0.0	0.0
mean	1.003004e+06	35.421207	0.420318	2.489290	9453.610858	NaN	NaN
std	1.716125e+03	12.754122	0.493632	1.115047	5222.355869	NaN	NaN
min	1.000001e+06	12.000000	0.000000	1.000000	188.000000	NaN	NaN
25%	1.001492e+06	27.000000	0.000000	1.500000	5443.000000	NaN	NaN
50%	1.003065e+06	33.000000	0.000000	2.000000	8109.000000	NaN	NaN
75%	1.004430e+06	43.000000	1.000000	3.000000	12675.000000	NaN	NaN
max	1.006040e+06	92.000000	1.000000	4.000000	23952.000000	NaN	NaN

```
[1] import pandas as pd
    import numpy as np
```

```
[2] df = pd.read_csv('Diwali Sales Data.csv', encoding= 'unicode_escape')
```

```
df.head()
```

ıame	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Category	Orders	Amount	Status	unnamed1
skriti	P00125942	F	26-35	28	0	Maharashtra	Western	Healthcare	Auto	1	23952.0	NaN	NaN
:artik	P00110942	F	26-35	35	1	Andhra Pradesh	Southern	Govt	Auto	3	23934.0	NaN	NaN
indu	P00118542	F	26-35	35	1	Uttar Pradesh	Central	Automobile	Auto	3	23924.0	NaN	NaN
ıdevi	P00237842	M	0-17	16	0	Karnataka	Southern	Construction	Auto	2	23912.0	NaN	NaN
Joni	P00057942	M	26-35	28	1	Gujarat	Western	Food Processing	Auto	2	23877.0	NaN	NaN

Next steps: Generate code with df | View recommended plots

Conclusion (Students should write in their own words):

Through this experiment we get to know about the different techniques of Data Analysis including Normalization and Kiscíctizatio→ı.

Post lab questions:

Q.1 What are some common challenges encountered during data cleaning? How did you handle missing values in the provided dataset?

Common Challenges:

- **Missing Data: Incomplete data entries are common and can arise from data entry errors or system issues.**
- **Outliers: Extreme values that don't conform to expected patterns can skew analysis.**
- **Inconsistent Data: Variations in data formats, units, or values that should be standardized.**
- **Duplicate Entries: Repeated data points that need to be identified and removed.**
- **Irrelevant Data: Presence of unnecessary or redundant features that don't contribute to the analysis.**

Handling Missing Values:

- **Deletion: If missing values are minimal, rows or columns can be removed.**
- **Imputation: Filling in missing data using methods like mean, median, or mode for numerical data, or using the most frequent value for categorical data.**
- **Advanced Techniques: Using algorithms like KNN (K-Nearest Neighbors) imputation, which predicts missing values based on the similarity with other data points.**

Q.2 Explain the importance of data normalization in the context of machine learning models. How does normalizing benefit the analysis?

Importance of Data Normalization:

- **Ensures Consistency: Many machine learning algorithms, such as gradient descent-based methods, are sensitive to the scale of input data. Features with larger ranges can disproportionately influence the model.**
- **Improves Model Performance: Normalizing data ensures that each feature contributes equally, leading to faster convergence during training and better overall model accuracy.**
- **Enhances Interpretability: Normalized data allows for easier interpretation and comparison of feature importance.**

Benefits to Analysis:

- **Avoiding Bias: Prevents any single feature from dominating the model due to its scale.**
- **Facilitating Faster Learning: Helps algorithms converge more quickly, improving efficiency.**
- **Boosting Accuracy: Leads to more accurate and reliable model predictions.**

Q.3 Discuss why it's essential to convert categorical variables like 'gender' into numerical representations.

Essentiality of Conversion:

- **Compatibility with Algorithms: Most machine learning algorithms require numerical input, so categorical variables must be converted to be processed correctly.**
- **Improved Model Performance: Converting categorical variables into numerical representations (e.g., one-hot encoding) allows models to treat these variables appropriately, enhancing prediction accuracy.**
- **Capturing Relationships: Numerical representations help in capturing relationships between different categories, especially in distance-based models like KNN or clustering algorithms.**

Example:

- **Gender Encoding: The categorical variable 'gender' can be converted into numerical values (e.g., 0 for 'male' and 1 for 'female') or one-hot encoded (e.g., [1,0] for 'male' and [0,1] for 'female') to ensure the model understands and utilizes this information effectively.**