Practical Experiment: Data Cleaning and Preprocessing in GCP

Objective:

To implement a Python script to read data and perform data cleaning and preprocessing steps using GCP.

Steps:

1. **Set up a GCP Environment**:
    - Create a GCP account if you don't have one.
    - Create a new project in GCP.
    - Enable the Google Cloud Storage and Google Compute Engine APIs.
2. **Create a Virtual Machine (VM) Instance**:
    - Go to the Google Compute Engine section.
    - Click on "Create Instance" to launch a new VM instance.
    - Choose the appropriate configuration (e.g., Ubuntu, n1-standard-1).
3. **Connect to the VM Instance**:
    - Use SSH to connect to your VM instance.
4. **Install Necessary Packages**:
    - Install Python and necessary libraries (Pandas, NumPy, etc.)

bash

```
sudo apt update
sudo apt install python3-pip
pip3 install pandas numpy
```

5. **Download the Dataset**:
    - Use the `wget` command to download the dataset and save it as `adult.csv`.

bash

```
wget https://archive.ics.uci.edu/ml/machine-learning-
databases/adult/adult.data -O adult.csv
```

6. **Implement the Data Cleaning and Preprocessing Script**:
    - Write the Python script to read data and perform data cleaning and preprocessing steps.

python

```
import pandas as pd
import numpy as np

# Step 1: Read Data
```

```
columns = ['age', 'workclass', 'fnlwgt', 'education',
'education_num', 'marital_status', 'occupation',
           'relationship', 'race', 'sex', 'capital_gain',
'capital_loss', 'hours_per_week', 'native_country', 'income']
data = pd.read_csv('adult.csv', header=None, names=columns,
na_values=' ?')

# Display the first few rows of the dataset
print("Original Data:\n", data.head())

# Step 2: Handle Missing Values
# Fill missing values with the most frequent value in each column
data = data.apply(lambda x: x.fillna(x.value_counts().index[0]))

# Display the dataset after handling missing values
print("\nData After Handling Missing Values:\n", data.head())

# Step 3: Remove Duplicates
# Remove duplicate rows
data.drop_duplicates(inplace=True)

# Display the dataset after removing duplicates
print("\nData After Removing Duplicates:\n", data.head())

# Step 4: Transform Data
# Apply a transformation to the 'capital_gain' column (e.g., log
transformation)
data['capital_gain'] = data['capital_gain'].apply(lambda x: np.log(x
+ 1))

# Display the dataset after transformation
print("\nData After Transformation:\n", data.head())

# Step 5: Replace Values
# Replace specific values in the 'income' column
data['income'] = data['income'].replace({' <=50K': 'Low', ' >50K':
'High'})

# Display the dataset after replacing values
print("\nData After Replacing Values:\n", data.head())

# Step 6: Detect and Filter Outliers
# Calculate the Z-scores to detect outliers
data['age_zscore'] = (data['age'] - data['age'].mean()) /
data['age'].std()

# Filter out rows where the Z-score is greater than 3 or less than -
3
data = data[(data['age_zscore'] < 3) & (data['age_zscore'] > -3)]

# Display the dataset after filtering outliers
print("\nData After Filtering Outliers:\n", data.head())

# Drop the Z-score column for final cleaned data
data.drop(columns=['age_zscore'], inplace=True)
```

```
# Display the final cleaned dataset
print("\nFinal Cleaned Data:\n", data.head())

# Save the cleaned data to a new CSV file
data.to_csv('cleaned_adult.csv', index=False)
print("\nCleaned data saved to 'cleaned_adult.csv'")
```

7. **Run the Script on GCP**:
    - Save the above script as `data_cleaning.py`.
    - Run the script using Python.

bash

```
python3 data_cleaning.py
```

Explanation:

1. **Set up GCP Environment**: Create and configure a GCP project, and enable necessary APIs.
2. **Create a Virtual Machine (VM) Instance**: Launch a new VM instance using Google Compute Engine.
3. **Connect to the VM Instance**: Use SSH to connect to your VM instance.
4. **Install Necessary Packages**: Install Python and the necessary libraries using pip.
5. **Download the Dataset**: Use the `wget` command to download the dataset and save it as `adult.csv`.
6. **Implement the Data Cleaning and Preprocessing Script**: The provided script reads the dataset, handles missing values, removes duplicates, transforms data, replaces values, and detects outliers. It saves the cleaned data to a new CSV file.
7. **Run the Script on GCP**: Execute the script on your GCP environment to perform data cleaning and preprocessing.