PRACTICAL-1

Apply data cleaning techniques on any dataset (e.g., Paper Reviews dataset in UCI repository). Techniques may include handling missing values, outliers and inconsistent values. A set of validation rules can be prepared based on the dataset and validations can be performed.

```python
import pandas as pd
import numpy as np

# Load dataset
df = pd.read_csv("/fruit_classification_dataset.csv")

# ------------ 1. Handling Missing Values ------------

# For numeric columns → replace missing values with median
num_cols = df.select_dtypes(include=['int64', 'float64']).columns
df[num_cols] = df[num_cols].fillna(df[num_cols].median())

# For categorical columns → replace missing values with mode
cat_cols = df.select_dtypes(include=['object']).columns
df[cat_cols] = df[cat_cols].fillna(df[cat_cols].mode().iloc[0])

# ------------ 2. Handling Outliers (Z-score method) ------------

for col in num_cols:
    mean = df[col].mean()
    std = df[col].std()
    z = (df[col] - mean) / std

    # Replace outliers (>3 or <-3) with median
```

```python
    # Replace outliers (>3 or <-3) with median
    df.loc[z > 3, col] = df[col].median()
    df.loc[z < -3, col] = df[col].median()

# ------------ 3. Fixing Inconsistent Values ------------

#  fruit_name column has inconsistent words
if 'fruit_name' in df.columns:
    df['fruit_name'] = df['fruit_name'].str.lower()

# size column might contain text variations
if 'size' in df.columns:
    df['size'] = df['size'].str.lower()

# ------------ 4. Validation Rules ------------
```

```python
rules = {}

# Rule 1: Weight must be positive
if 'weight' in df.columns:
    rules['weight_valid'] = (df['weight'] > 0).sum()

# Rule 2: Fruit names must be one of known fruits
if 'fruit_name' in df.columns:
    valid_list = ['apple', 'banana', 'mango', 'orange', 'grape']
    rules['fruit_name_valid'] = df['fruit_name'].isin(valid_list).sum()

# Rule 3: Size must be small/medium/large
if 'size' in df.columns:
    rules['size_valid'] = df['size'].isin(['small', 'medium', 'large']).sum()

print("\n--- Validation Summary ---")
print(rules)

# ------------ 5. Save cleaned dataset ------------
df.to_csv("fruits_cleaned.csv", index=False)

print("\nData cleaning completed!")
```

```
--- Validation Summary ---
{'fruit_name_valid': np.int64(2480)}

Data cleaning completed!
```