Outlier Detection and Imputation Documentation

# 1. Data Preparation

- Load the Data: Import the dataset into your environment (e.g., using `pandas` in Python).  
 - Initial Inspection:  
 - Check for any missing values.  
 - Analyze data types of each feature.  
 - Generate basic descriptive statistics to understand the data distribution (mean, median, standard deviation, etc.).

```python  
import pandas as pd  
  
# Load data  
df = pd.read\_csv('data.csv')  
  
# Check data info and summary  
print(df.info())  
print(df.describe())  
```

# 2. Detecting Outliers

## 2.1 Using Statistical Methods

- Z-score Method:  
 - Calculate the z-score for each data point using `scipy.stats.zscore`.  
 - Identify data points with z-scores exceeding a chosen threshold (e.g., >3 or <-3) as outliers.

# Select only the numerical columns

numerical\_df = df.select\_dtypes(include=[np.number])

# Calculate the Z-scores for each numerical column

z\_scores = np.abs(stats.zscore(numerical\_df))

# Set a threshold for identifying outliers

threshold = 3

# 3. Outlier Imputation

Once outliers are detected, handle them with suitable imputation methods:

## 3.1 Imputation Techniques

- Mean/Median Imputation: Replace outliers with the mean or median of the column. This is common when outliers are sparse.  
- Mode Imputation: Replace outliers with the mode, especially useful for categorical features.

# Assuming you want to replace outliers based on z-score

# and replace them with median for the column

upper\_limit = numerical\_df['MSSubClass'].mean() + threshold \* numerical\_df['MSSubClass'].std()

median = numerical\_df['MSSubClass'].median()

df['MSSubClass'] = df['MSSubClass'].apply(lambda x: median if x > upper\_limit else x)

print(df['MSSubClass'])

# 4. Validation

- Post-Imputation Analysis:

- Verify that the imputed values have reduced or eliminated outliers.  
# Recalculate the Z-scores after imputation

z\_scores\_post\_imputation = np.abs(stats.zscore(df[['MSSubClass']]))

# Identify outliers after imputation

outliers\_after = (z\_scores\_post\_imputation > threshold).sum(axis=0)

# Print out the number of outliers before and after imputation

print(f"Outliers before imputation: {df['MSSubClass']}")

print(f"Outliers after imputation: {outliers\_after}")