1. **Generate Python Code: Data Quality Report Generation:**

You are a Python programmer tasked with automating the generation of a comprehensive data quality report. Write Python code that performs the following:

1. **Input**: Load the provided dataset (CSV file) with 'Y' as the target variable.
2. **Analysis**:
   * **For Numerical Features**: Compute and display the following statistics:
     + Percentage of Missing Values
     + Cardinality (Number of unique values)
     + Minimum Value
     + 1st Quartile (25th percentile)
     + Mean Value
     + Median Value
     + 3rd Quartile (75th percentile)
     + Maximum Value
     + Standard Deviation
   * **For Categorical Features**: Compute and display the following statistics:
     + Percentage of Missing Values
     + Cardinality (Number of unique categories)
     + Mode (Most frequent category)
     + Frequency of the Mode
     + Percentage Frequency of the Mode
     + Second Mode (Second most frequent category)
     + Frequency of the Second Mode
     + Percentage Frequency of the Second Mode
3. **Output**:
   * Present the results in a clear, tabular format (e.g., pandas DataFrame or print statements).
   * Save the data quality report to a CSV file.

**Deliverables:**

Provide the complete Python code to perform the above tasks, ensuring clarity and proper documentation within the script.

1. **Generate Python Code: Data Quality Issue Identification and Handling Strategy:**

You are a Python programmer tasked with identifying data quality issues and proposing handling strategies. Write Python code that performs the following:

1. **Input**: Load the dataset (CSV file) and reference the data quality report generated previously.
2. **Identification**:
   * Detect and flag data quality issues, including:
     + **Missing Values**: Identify columns with missing values and calculate their percentages.
     + **Outliers**: Detect numerical outliers using statistical thresholds (e.g., values beyond 3 standard deviations).
     + **Irregular Cardinality**: Identify columns with unusually high or low cardinality for their data type.
   * For each identified issue, associate the affected feature and describe the issue.
3. **Handling Strategy**:
   * For each issue, propose a handling strategy, such as:
     + Impute missing values (e.g., mean/median/mode).
     + Transform or remove outliers.
     + Group or consolidate categories with low frequency.
4. **Output**:
   * Generate a table (pandas DataFrame) with the following columns:
     + **Feature Name**
     + **Issue**
     + **Handling Strategy**
   * Save this table as a CSV file.

**Deliverables:**

Provide the Python code to automate this process, ensuring proper comments and documentation to explain the logic behind the steps.

1. **Generate Python Code: Apply Data Quality Handling Strategies:**

You are a Python programmer tasked with applying handling strategies to improve dataset quality. Write Python code that performs the following:

1. **Input**:
   * Load the dataset (CSV file).
   * Load the data quality issues and handling strategies table generated previously.
2. **Processing**:
   * For each feature listed in the handling strategy table:
     + **Missing Values**: Apply the specified imputation method (e.g., mean, median, mode) or remove rows/columns as needed.
     + **Outliers**: Handle outliers based on the recommended approach (e.g., capping, removal).
     + **Irregular Cardinality**: Group or consolidate categories with low frequency as specified.
3. **Validation**:
   * Verify that all issues have been addressed appropriately.
   * Ensure the cleaned dataset is consistent and free from the identified issues.
4. **Output**:
   * Save the cleaned dataset to a new CSV file.
   * Provide a summary of the changes made, detailing the handling strategy applied for each feature.

**Deliverables:**

Provide the Python code to automate this process, ensuring the script is modular, well-documented, and easily reusable.

1. **Generate Python Code: Dataset Cleaning Instructions (Dataset 1)**

You are a Python programmer tasked with cleaning a dataset based on specific instructions. Write Python code that performs the following operations on the attached CSV dataset with 'Y' as the target feature.

**Instructions:**

1. **Data Cleaning Tasks**:
   * **Unify Units**: Standardize the units for the following columns: Weight, Memory, Inches.
   * **Split Columns**:
     + Split the Memory column into two columns: Size and Drive Type.
     + Split the ScreenResolution column into two columns: Resolution and Display Type.
   * **Reduce Cardinality**: Group entries in the following columns to reduce cardinality: GPU, CPU. Use logical or domain-specific grouping (e.g., based on brand or performance categories).
2. **Validation**:
   * Verify the changes to ensure all specified transformations are correctly applied.
   * Ensure the dataset integrity is maintained, with no unintended alterations.
3. **Output**:
   * Save the cleaned dataset as a new CSV file (cleaned\_dataset1.csv).
   * Print a summary of the transformations applied to the dataset.

**Deliverables:**

Provide Python code to automate this process, ensuring modularity, proper error handling, and detailed comments to explain each step.

1. **Generate Python Code: Dataset Cleaning Instructions (Dataset 2)**

You are a Python programmer tasked with cleaning a dataset based on specific instructions. Write Python code to perform the following operations on the attached CSV dataset with 'Y' as the target feature.

**Instructions:**

1. **Data Cleaning Tasks**:
   * **Splitting or merging columns**
   * **Standardizing units**
   * **Normalization**
   * **Reducing cardinality for categorical variables**
   * **Renaming columns for clarity or consistency**
2. **Validation**:
   * Check if all transformations are applied correctly without altering the integrity of the data.
   * Ensure that the target feature 'Y' remains unaffected during the cleaning process.
3. **Output**:
   * Save the cleaned dataset to a new CSV file (cleaned\_dataset2.csv).
   * Provide a summary of the cleaning operations performed.

**Deliverables:**

Provide Python code that automates the data cleaning process, ensuring each task is properly handled with clear comments and error checks.

1. **Generate Python Code: Dataset Splitting: Training and Testing Sets [Both Datasets]**

You are a Python programmer tasked with preparing data for machine learning. Write Python code to split the dataset into training and testing sets.

**Instructions:**

1. **Input**:
   * Load the dataset (CSV file) with 'Y' as the target feature.
2. **Split Criteria**:
   * Use an **80/20 split** to create training and testing datasets.
   * Ensure the split is **stratified** based on the target feature 'Y' to maintain class distribution across both sets.
   * Use a **fixed random state** for reproducibility.
3. **Output**:
   * Save the training and testing sets as separate CSV files (training\_set.csv and testing\_set.csv).
   * Print a summary showing the number of rows in each set and the class distribution of 'Y' in both sets.

**Deliverables:**

Provide Python code that automates this process using libraries such as pandas and scikit-learn. Ensure the code is well-documented and easy to follow.

1. **Generate Python Code: Train a Machine Learning Model (Dataset 1, 2) [change the Metrics information]**

You are a Python programmer tasked with training a machine learning model using the provided training dataset. Write Python code that performs the following:

**Instructions:**

1. **Input**:
   * Load the training dataset (CSV file) with 'Y' as the target feature.
2. **Model Training**:
   * Use the specified algorithm [Insert Algorithm Here] to train the model.
   * Perform **5-fold cross-validation** on the training data to evaluate the model's performance.
3. **Evaluation**:
   * Calculate and display the following metrics for the model on the training set:
     + **R-Squared**
     + **Adjusted R-Squared** (if applicable)
     + **Mean Squared Error (MSE)**
     + Additional metrics as relevant to the algorithm or problem type (e.g., accuracy for classification tasks).
4. **Output**:
   * Save the trained model to a file (e.g., using joblib or pickle).
   * Print a summary of the model’s performance across all folds, including the mean and standard deviation of the metrics.

**Deliverables:**

Provide Python code to automate this process, ensuring clarity, proper modularization, and in-code comments to explain each step.

1. **Generate Python Code: Test the Machine Learning Model (Dataset 1,2) [change the Metrics information]**

You are a Python programmer tasked with evaluating a machine learning model on unseen data. Write Python code to test the model using the provided testing dataset.

**Instructions:**

1. **Input**:
   * Load the testing dataset (CSV file) with 'Y' as the target feature.
   * Load the trained model from the file created in the training step.
2. **Model Testing**:
   * Use the trained model to make predictions on the testing dataset.
3. **Evaluation**:
   * Compute and report the following metrics for the testing dataset:
     + **R-Squared**
     + **Adjusted R-Squared** (if applicable)
     + **Mean Squared Error (MSE)**
     + Additional relevant metrics based on the problem type (e.g., RMSE, MAE for regression; accuracy, precision, recall for classification).
4. **Analysis**:
   * Compare testing performance metrics with training metrics to assess the model's generalization ability.
   * Provide insights into potential overfitting or underfitting based on the results.
5. **Output**:
   * Print the evaluation metrics for the testing dataset.
   * Save the predictions alongside the actual values in a new CSV file (model\_predictions.csv).

**Deliverables:**

Provide Python code to automate this process, ensuring readability and proper documentation of each step in the code.