Data Preprocessing (heart disease)

== Step 1 ==

Use the following information about the heart disease dataset to provide me with Python code snippets for handling multiple forms of dirty data. I will provide you with a series of instruction prompts for generating the code, one by one, after this information.

Dataset Description

The dataset contains 14 variables: 13 independent variables representing potential risk factors for heart disease and one dependent variable indicating the diagnosis.

Dependent Variable

- **Num:** Diagnosis result, represented as categorical data: **(1**: Presence of heart disease, **0**: Absence of heart disease)

Independent Variables

- **Age**: Patient age in years (integer)
- **Sex**: Binary categorical data (1: Male, 0: Female)
- **Chest Pain Type (cp)**: Categorical data with four types (1: Typical angina, 2: Atypical angina, 3: Non-anginal pain, 4: Asymptomatic)
- **Resting Blood Pressure (trestbps):** Patient's resting blood pressure (mm Hg) (integer)
- **Serum Cholesterol (chol)**: Serum cholesterol levels (mg/dl) (integer)
- **Fasting Blood Sugar (fbs)**: Binary categorical data (1: >120 mg/dl, 0: ≤120 mg/dl)
- **Resting Electrocardiographic Results (restecg)**: Categorical data with three types (0: Normal, 1: ST-T wave abnormality, 2: Left ventricular hypertrophy)
- **Maximum Heart Rate Achieved (thalach)**: Highest heart rate during testing (integer)
- **Exercise-Induced Angina (exang)**: Binary categorical data (1: Yes, 0: No)
- **ST Depression (oldpeak)**: ST depression induced by exercise (float)
- **Slope of Peak Exercise ST Segment (slope)**: Categorical data with three types (1: Upsloping, 2: Flat, 3: Downsloping)
- **Number of Major Vessels (ca)**: Number of major vessels colored by fluoroscopy (integer ranging from 0 to 3)

- **Thalassemia (thal)**: Categorical data with three types (3: Normal, 6: Fixed defect, 7: Reversible defect)

Sample Data

Example records from the dataset:

- {age, sex, cp, trestbps, chol, fbs, restecg, thalach, exang, oldpeak, slope, ca, thal, num}
- {63, 1, 1, 145, 233, 1, 2, 150, 0, 2.3, 3, 0, 6, 0}
- {67, 1, 4, 160, 286, 0, 2, 108, 1, 1.5, 2, 3, 3, 2}
- {67, 1, 4, 120, 229, 0, 2, 129, 1, 2.6, 2, 2, 7, 1}

== Step 2: Dealing with missing values ==

- Use the dataset from the file "HeartDisease.csv" and provide me with a Python code snippet to detect and delete "NULLs" or "blanks" in the column "ca," representing the number of major vessels, and the column "thal," representing Thalassemia in the dataset. This Python code snippet must output the record(s) containing NULLs or blanks. The remaining dataset, excluding the NULLs and blanks, will be saved in a file named **"HeartDisease_preprocessed1.csv,"** which will be ready for processing in the next step.

== Step 3: Dealing with erroneous data ==

- Use the dataset from the file "HeartDisease_preprocessed1.csv" and provide me with a Python code snippet to handle the erroneous data in the column "num." A value in this column must be either 0 or 1; the other values (i.e., 2, 3, and 4) found in this column are errors. This Python code snippet must convert the errors to 1. After correcting the errors, the entire dataset will be saved in a file named **"HeartDisease_preprocessed2.csv,"** which will be ready for processing in the next step.

== Step 4: Dealing with outliers ==

- Use the dataset from the file **"HeartDisease_preprocessed2.csv"** and provide a Python code snippet to implement a **boxplot**, perform **Z-score** analysis, and calculate the **interquartile range** (IQR) to identify potential outliers in the column **"age"**. The Python code snippet must output the outliers filtered from the dataset. The dataset, with outliers removed, will be saved in a file named **"HeartDisease_preprocessed3.csv"**.

- Use the dataset from the file **"HeartDisease_preprocessed3.csv"** and provide a Python code snippet to implement a **boxplot**, perform **Z-score** analysis, and calculate the **interquartile range** (IQR) to identify potential outliers in the column **"trestbps"**, which contains integers representing the patient's resting blood pressure (mm Hg). The Python code snippet must output the outliers filtered from the dataset. The dataset, with outliers removed, will be saved in a file named **"HeartDisease_preprocessed4.csv"**.

- Use the dataset from the file **"HeartDisease_preprocessed4.csv"** and provide a Python code snippet to implement a **boxplot**, perform **Z-score** analysis, and calculate the interquartile range (IQR) to identify potential outliers in the column **"chol"**, which **contains** integers representing serum cholesterol levels (mg/dl). The **Python** code snippet must output the outliers filtered from the dataset. The dataset, with outliers removed, will be saved in a file named **"HeartDisease_preprocessed5.csv"**.

- Use the dataset from the file **"HeartDisease_preprocessed5.csv"** and provide a Python code snippet to implement a **boxplot**, perform **Z-score** analysis, and calculate the interquartile range (IQR) to identify potential outliers in the column **"thalach"**, which contains integers representing the highest heart rate recorded during testing. The Python code snippet must output the outliers filtered from the dataset. The dataset, with outliers removed, will be saved in a file named **"HeartDisease_preprocessed6.csv".**

== Step 5: Data Normalization ==

- Provide a Python code snippet to convert numerical data to categorical data in the column **“chol”**. The code snippet must first check the integers in this column. If an integer is less than 200, convert it to 1, indicating that the patient’s cholesterol level is desirable. If an integer is equal to or greater than 200 but not over 239, convert it to 2, indicating that the patient’s cholesterol level is at the borderline of a high level. If an integer is equal to or greater than 240, convert it to 3, indicating that the patient’s cholesterol level is high. This Python code snippet will process data in the file named **“HeartDisease_preprocessed6.csv”**. After converting the data in the column **“chol”**, the entire dataset will be saved in a new file named **“HeartDisease_preprocessed7.csv”**.

- Provide a Python code snippet to convert numerical data to categorical data in the column **“trestbps.”** The code snippet must first check the integers in this column. If an integer is less than or equal to 129, convert it to 0, indicating that the patient's systolic BP is not at risk of heart disease. If an integer is equal to or greater than 130, convert it to 1, indicating that the patient's systolic BP is at risk of heart disease. This Python code snippet will process data in the file named **“HeartDisease_preprocessed7.csv.”** After converting the data in the **“trestbps”** column, the entire dataset will be saved in a new file named **“HeartDisease_preprocessed8.csv.”**

- Provide a Python code snippet to convert numerical data to categorical data in the column **“thalach.”** The code snippet must first retrieve an integer from the column **“age”** and then calculate (220−age) × 0.85. If the result of the calculation is equal to or greater than an integer in the column **“thalach,”** convert this integer to 0, indicating that the patient is not at risk of heart disease. If the result of the calculation is less than an integer in the column **“thalach,”** convert this integer to 1, indicating that the patient is at risk of heart disease. This Python code snippet will process data in the file named **“HeartDisease_preprocessed8.csv.”** After converting all data in the “thalach” column, the entire dataset will be saved in a new file named **“HeartDisease_preprocessed9.csv.”**

== Step 6: Feature Engineering ==

- Provide a Python code snippet to implement feature engineering where the columns **"sex"** and **"age"** are combined to create a new column named **"gender_and_age"**. The data in the new column are categorical, containing integers 1–4:
    - If the value in the column **"sex" is 1 (male)** and the value in the column **"age" is less than 45**, assign **1** to the column "gender_and_age", indicating that a male patient whose age is below 45 years is not at risk of heart disease.
    - If the value in the column **"sex" is 1 (male)** and the value in the column **"age" is equal to or greater than 45**, assign **2** to the column "gender_and_age", indicating that a male patient whose age is 45 years or older is at risk of heart disease.
    - If the value in the column **"sex" is 0 (female)** and the value in the column **"age" is less than 55**, assign **3** to the column "gender_and_age", indicating that a female patient whose age is below 55 years is not at risk of heart disease.
    - If the value in the column **"sex" is 0 (female)** and the value in the column **"age" is equal to or greater than 55**, assign **4** to the column "gender_and_age", indicating that a female patient whose age is 55 years or older is at risk of heart disease.
    - This Python code snippet will process data from the file named **“HeartDisease_preprocessed9.csv.”** After finishing the creation of the "gender_and_age" column, the entire dataset will be saved in a new file named **“HeartDisease_preprocessed10.csv.”**

- Provide a Python code snippet to implement feature engineering where data in the two columns, **“slope”** and **“oldpeak”**, are combined to create a new column named **“slope_and_oldpeak”**. The data in the new column are categorical, containing integers **0** or **1**.

  - If the value in the column **“slope”** is **1**, assign **0** to the column **“slope_and_oldpeak”**, indicating that the patient is not at risk of heart disease.
  - If the value in the column **“slope”** is **2** or **3**, and the value in the column **“oldpeak”** is less than **1**, assign **0** to the column **“slope_and_oldpeak”**, indicating that the patient is not at risk of heart disease.
  - If the value in the column **“slope”** is **2** or **3**, and the value in the column **“oldpeak”** is equal to or greater than **1**, assign **1** to the column **“slope_and_oldpeak”**, indicating that the patient is at risk of heart disease.

- This Python code snippet will process data in a file named **“HeartDisease_preprocessed10.csv”**. After finishing creating the data in the **"slope_and_oldpeak"** column, the entire dataset will be saved in a new file named **“HeartDisease_preprocessed11.csv”**.