Disclaimer and caution: This script/file was generated via generative AI

Thursday February 13, 2025, without prejudice

Approach:

Missing data can be handled as follows:

1. For missing categorical or discrete data, replace it with the most frequent value.
2. For missing continuous data, use the mean or median value.
3. If it is the target attribute, drop the entire row.

Prompt:

'Write a multiline formatted Python code that reads a csv record from a URL and does the following.

1. Identify the columns which have values as "?" in them.
2. Replace these with the mean value of the respective attribute.
3. Modify the data type of the attribute to float after replacement. The code should adhere to recent programming practices, and include clear variable names, and being structured for readability and maintainability.'

Response:

Certainly! Below is a well-structured and documented Python code that reads a CSV record from a URL, identifies columns with "?" values, replaces them with the mean of the respective columns, and converts the columns to float data type. The code includes error handling and follows consistent coding practices.

```
import pandas as pd
import numpy as np

def read_and_clean_csv(url):
    """
    Reads a CSV file from a given URL, replaces "?" with the mean of the
respective columns,
    and converts those columns to float type.

    Parameters:
        url (str): The URL of the CSV file to read.

    Returns:
        pd.DataFrame: A DataFrame with cleaned data.
    """
    try:
        # Read the CSV file from the provided URL
        data = pd.read_csv(url)

        # Identify columns with "?" and replace them with NaN for easier
processing
        data.replace("?", np.nan, inplace=True)
        # Iterate through each column to replace NaN values with the mean
        for column in data.columns:
            if data[column].isnull().any():  # Check if there are NaN values in the column
                # Calculate the mean of the column, ignoring NaN values
                mean_value = data[column].astype(float).mean()

                # Replace NaN values with the calculated mean
                data[column].fillna(mean_value, inplace=True)

                # Convert the column to float type
                data[column] = data[column].astype(float)

        return data

    except Exception as e:
        # Handle exceptions and provide informative error messages
        print(f"An error occurred: {e}")
        return None

# Example usage
if __name__ == "__main__":
    url = "https://example.com/data.csv"  # Replace with the actual CSV URL
    cleaned_data = read_and_clean_csv(url)

    if cleaned_data is not None:
        print("Cleaned Data:")
        print(cleaned_data.head())
```

Explanation of the Code:

```
Imports: pandas for data manipulation and numpy for handling NaN values.
Function read_and_clean_csv: This function takes a URL as input, reads the CSV file, and cleans the data.
    It uses pd.read_csv to read the CSV file.
    It replaces "?" with NaN for easier handling of missing values.
    It checks each column for NaN values, calculates the mean, and replaces NaN with the mean.
    Finally, it converts the column data type to float.
Error Handling: A try-except block is included to catch and print any errors that occur during the execution of the function.
Example Usage: The code includes a main block to demonstrate how to use the function, which can be modified with the actual URL of the CSV file.
```

This structure ensures that the code is readable, maintainable, and adheres to recent programming practices.

Here's a simpler script response:

```
import pandas as pd
# Read CSV record from URL
url = 'https://example.com/data.csv'
data = pd.read_csv(url)
print(data.dtypes)
# Identify columns with "?" values
cols_with_question_mark = data.columns[data.isin(['?']).any()]
print(cols_with_question_mark)
# Replace "?" values with the mean value of the respective attribute
for col in cols_with_question_mark:
    mean_val = pd.to_numeric(data[col], errors='coerce').mean()
    data[col] = pd.to_numeric(data[col].replace('?', mean_val))
# Modify the data type of the attribute to float after replacement
data[cols_with_question_mark] = data[cols_with_question_mark].astype(float)
# Print the modified data
print(data.dtypes)
```