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

Thursday February 13, 2025, without prejudice

Approach:

Always good to check for spurious outliers.

Prompt:

'Write a python code to extract entries of a dataframe where the attribute 'Price' has outliers that might be anomalies in comparison to the other data.'

Response:

To identify outliers in a DataFrame based on the 'Price' attribute, we can use the Interquartile Range (IQR) method, which is a common statistical technique for detecting anomalies. Below is a well-structured Python code snippet that demonstrates how to extract entries with outliers in the 'Price' column of a DataFrame.

```
import pandas as pd

def extract_price_outliers(df, column='Price'):
    """
    Extract entries from the DataFrame where the specified column contains outliers.

    Parameters:
    df (pd.DataFrame): The input DataFrame containing the data.
    column (str): The name of the column to check for outliers. Default is 'Price'.

    Returns:
    pd.DataFrame: A DataFrame containing only the outlier entries.
    """
    # Check if the specified column exists in the DataFrame
    if column not in df.columns:
        raise ValueError(f"Column '{column}' does not exist in the DataFrame.")

    # Calculate Q1 (25th percentile) and Q3 (75th percentile)
    Q1 = df[column].quantile(0.25)
    Q3 = df[column].quantile(0.75)

    # Calculate the Interquartile Range (IQR)
    IQR = Q3 - Q1

    # Define the lower and upper bounds for outliers
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR

    # Extract entries that are considered outliers
    outliers = df[(df[column] < lower_bound) | (df[column] > upper_bound)]

    return outliers

# Example usage
if __name__ == "__main__":
    # Sample DataFrame creation for demonstration
    data = {
        'Product': ['A', 'B', 'C', 'D', 'E', 'F', 'G'],
        'Price': [10, 12, 12, 13, 100, 15, 14]
    }

    df = pd.DataFrame(data)

    # Extracting outliers from the 'Price' column
    try:
        outlier_entries = extract_price_outliers(df)
        print("Outlier Entries:")
        print(outlier_entries)
    except ValueError as e:
        print(e)
```

The example output:

```
    Product  Price
4       E    100
```

Regenerate response:

```
import pandas as pd

# Read CSV record from URL
url = "https://example.com/replace-with-own-data/laptop_pricing_dataset_base.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)

#print("Total number of rows before removal:", len(data))

# Remove duplicate entries
data.drop_duplicates(inplace=True)

# Print the total number of rows after removal
#print("Total number of rows after removal:", len(data))

import numpy as np

# Assuming 'data' is the pandas DataFrame containing the data

# Calculate the mean and standard deviation of the prices
mean_price = data['Price'].mean()
std_price = data['Price'].std()

# Define a threshold for outliers (e.g., 3 standard deviations away from the mean)
outlier_threshold = 3

# Identify outliers using the threshold
outliers = data[(np.abs(data['Price'] - mean_price) > outlier_threshold *
std_price)]

# Extract entries with price outliers
entries_with_outliers = data[data['Price'].isin(outliers['Price'])]

# Print the entries with price outliers
print("Entries with price outliers:")
print(entries_with_outliers)
```

Output

```
Entries with price outliers:
    Manufacturer  Category     Screen  GPU  OS  CPU_core  Screen_Size_cm  \
64          Asus         1    Full HD    3   1         7          43.942
77          Dell         5    Full HD    3   1         7          43.942
144       Lenovo         3  IPS Panel    3   1         7          43.180
159        Razer         1    Full HD    3   1         7          35.560

     CPU_frequency  RAM_GB  Storage_GB_SSD  Weight_kg  Price
64             2.9      16             256       3.60   3810
77             2.9      16             256       3.42   3665
144            2.8       8             256       3.40   3810
159            2.8      16             256       1.95   3301
```