Raghunath Singh

Elevate Labs Internship

Task 1: Data Cleaning and Pre-processing

GitHub Repository Link:

https://github.com/RaghunathSinghOfficial/elevate_Labs_DA_Internship_task1_Superstore_Sales

---

Elevate Data Analyst Internship - Interview Questions: Task 1 Reflection

---

1. What are missing values and how do you handle them?

Missing values are simply absent data points in a dataset, which can appear as blank cells, null, NaN, or specific placeholders like "N/A". They occur for various reasons, such as data entry errors, data corruption, or information not being collected.

In my recent Task 1 with the Superstore Sales Data, I identified missing values using Power Query's **"Column Quality"** and **"Column Distribution"** features. For this particular dataset, I found 0% empty values, so no specific handling was required.

However, if I had found missing values, I would handle them by either removing the rows/columns if the missing data is minimal and non-critical, or by imputing/replacing them with a suitable value (e.g., mean, median, mode, or a placeholder like "Unknown") if removal would cause significant data loss. In Power Query, I use "Remove Empty" or "Replace Values."

2. How do you treat duplicate records?

I treat duplicate records by identifying and removing them to ensure data uniqueness and prevent skewed analysis. I decide if uniqueness applies across all columns or a specific subset. In Power Query, I use the "Remove Duplicates" function after selecting the relevant column(s) or the entire table.

3. Difference between dropna() and fillna() in Pandas?

While I used Excel and Power Query for this task, I'm familiar with Python's Pandas library for data manipulation, and dropna() and fillna() are core functions for handling missing values there:

- **dropna():** This function is used to **drop (remove) rows or columns that contain missing values (NaN)**.
  - df.dropna(): Removes rows with *any* missing values.
  - df.dropna(how='all'): Removes rows only if *all* values in that row are missing.
  - df.dropna(axis=1): Removes columns with missing values.
  - It's like Power Query's "Remove Empty Rows" feature, but with more fine-grained control over rows vs. columns and conditions.
- **fillna():** This function is used to **fill (impute) missing values** with a specified value or method (e.g., mean, median, forward-fill).
  - df.fillna(value=0): Replaces all NaN values with 0.
  - df.fillna(method='ffill'): Fills NaN values with the last valid observation (forward fill).
  - df.fillna(df.mean()): Fills NaN values in each column with that column's mean.
  - This is conceptually similar to Power Query's "Replace Values" or "Fill Down/Up" options, offering powerful imputation capabilities.

4. What is outlier treatment and why is it important?

Outlier treatment involves identifying and managing data points that significantly deviate from the norm. It's important because outliers can heavily distort statistical analyses, skew model training, and mislead visualizations, leading to inaccurate insights. Treatment can involve removal, transformation, or capping.

5. Explain the process of standardizing data.

Standardizing data is about transforming it into a consistent, uniform format. For text, this means ensuring consistent casing or replacing variations (e.g., "USA" vs. "U.S.A.") through formatting options or find-and-replace. For numerical data, it often means scaling values to a common range (e.g., Z-score normalization) to prevent certain features from dominating analysis.

6. How do you handle inconsistent data formats (e.g., date/time)?

I handle inconsistent data formats by explicitly converting them to a uniform type. For dates, particularly MM/DD/YYYY formats from diverse sources, I specifically use Power Query's "**Using Locale...**" option, setting the data type to Date and the locale to English (United States). This ensures correct interpretation and consistent formatting.

7. What are common data cleaning challenges?

Common data cleaning challenges include **inconsistent data types** (date/number formats)**, missing values, duplicate records, inconsistent text formatting/spellings** (e.g., "first class", "First Class")**, outliers,** and **structural errors (**like misaligned columns, header rows mixed with data**)**. Each requires a specific approach to ensure data quality.

8. How can you check data quality?

I check data quality using several methods:

- **Profiling tools** (like Power Query's "Column Quality" and "Column Distribution") for an overview of completeness and distribution.
- **Visual inspection** and **filtering/sorting** to spot inconsistencies.
- **Summary statistics** to identify unexpected values.
- Unique value counts for categorical consistency.

These checks help systematically identify issues.