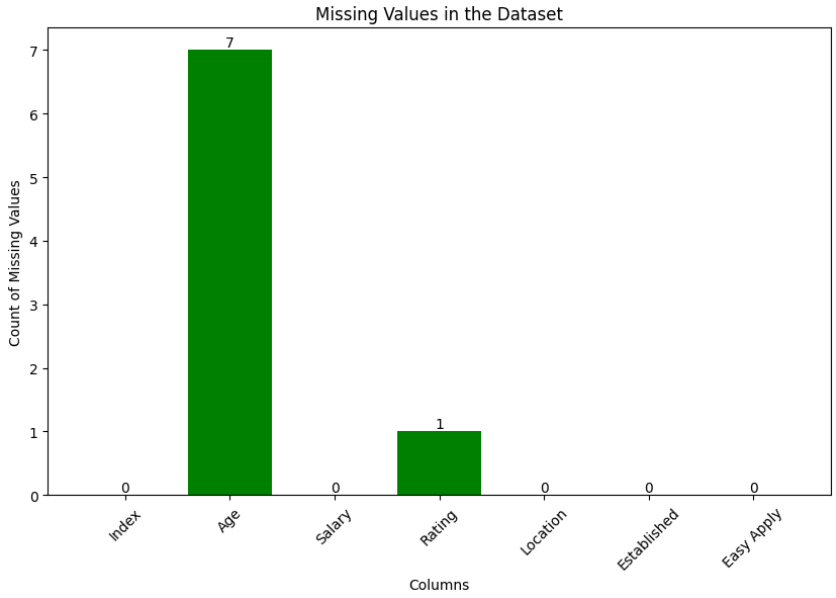
PrepInsta

Data Quality Assessment and Preprocessing

1. Missing Values:

Question**:** Are there any missing values in the dataset, and if so, how should they be handled for each indicator?

The dataset contains missing values in the 'Age' and 'Rating' columns, with 7 and 1 missing entries respectively.



2. Data Types:

Question: What are the data types of each indicator, and do they align with their expected types (e.g., numerical, categorical)?

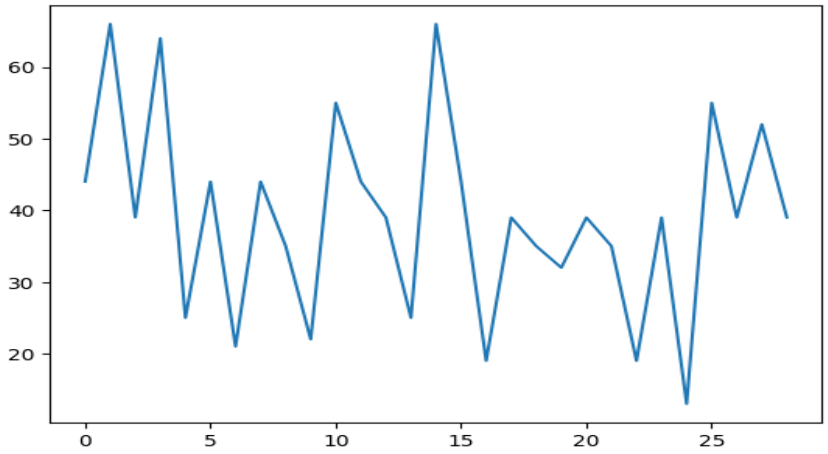
The dataset's columns have the following data types: Index is 'int64', Age is 'float64', Salary is 'object', Rating is 'float64', Location is 'object', Established is 'int64', and Easy Apply is 'object'. While most columns align with their expected types, further examination is needed for the 'Salary', 'Location', and 'Easy Apply' columns to ensure they match their intended data types for analysis.

```
Data types of each column:
Index            int64
Age            float64
Salary          object
Rating         float64
Location        object
Established      int64
Easy Apply      object
dtype: object
```

3. Outliers:

Question: Identify potential outliers in numerical indicators (e.g., Age, Salary, Rating). Should outliers be removed or adjusted?

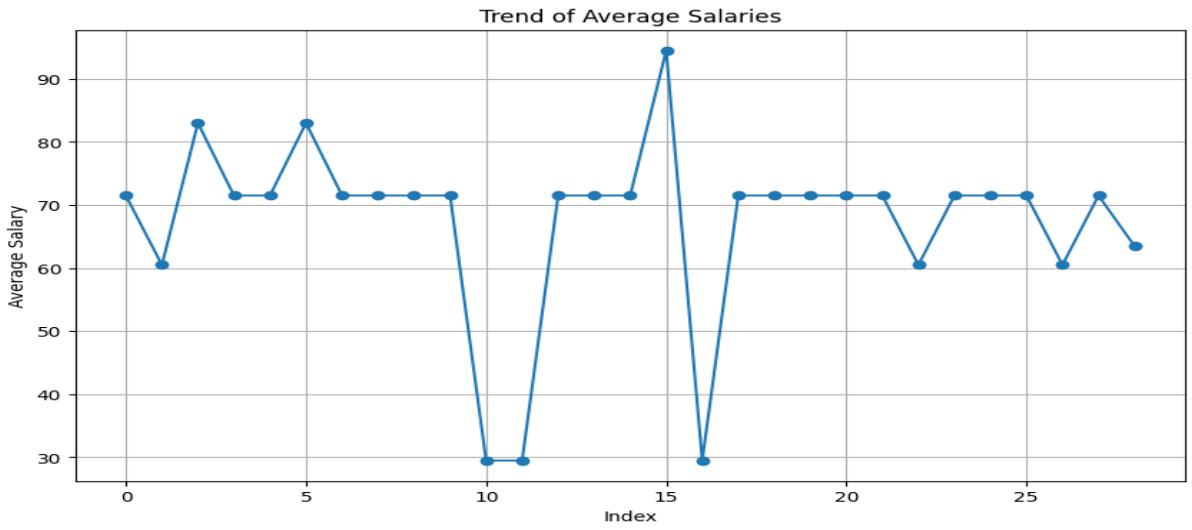
- Calculate the mean age value: The mean age value is calculated as the average of the available ages in the dataset.
- Fill missing age values with the mean: Any missing age values in the dataset are filled with the calculated mean age.
- Round age values: The age values are rounded to one decimal place for consistency.



4. Salary Formatting:

Question: Examine the format of the Salary column. Does it require any formatting or standardization for consistent analysis?

The Salary column is processed to ensure a consistent format for analysis. It begins by removing special characters like "$" and "k" using regular expressions. The salary range is then split into minimum and maximum values, which are converted to integers. A new column for the average salary is calculated as the midpoint between the minimum and maximum values. These steps ensure uniform formatting for easier comparison of salary ranges within the dataset.



5. Location Standardization:

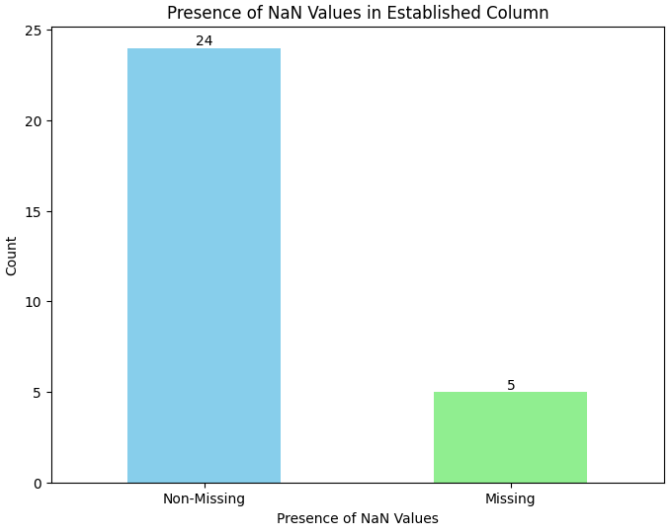
Question: Check the consistency of location entries. Do they need standardization, and how can this be achieved?

The location entries are standardized by mapping variations to their standard forms using a function. For example, 'India,In' and 'India In' are both mapped to 'India', while 'New York,Ny' is mapped to 'New York', and 'Australia Aus' is mapped to 'Australia'. This ensures consistency in the location data for analysis.

6. Established Column:

Question: Explore the Established column. Are there any inconsistencies or anomalies that need to be addressed?

In examining the Established column, it was found that certain entries were represented as -1, which is not a valid establishment year. To address this inconsistency, all instances of -1 in the Established column were replaced with NaN, signifying missing or unknown values. This cleaning step ensures that the Established column now contains consistent and accurate data, free from invalid entries that could impact the analysis.



7. Easy Apply Indicator:

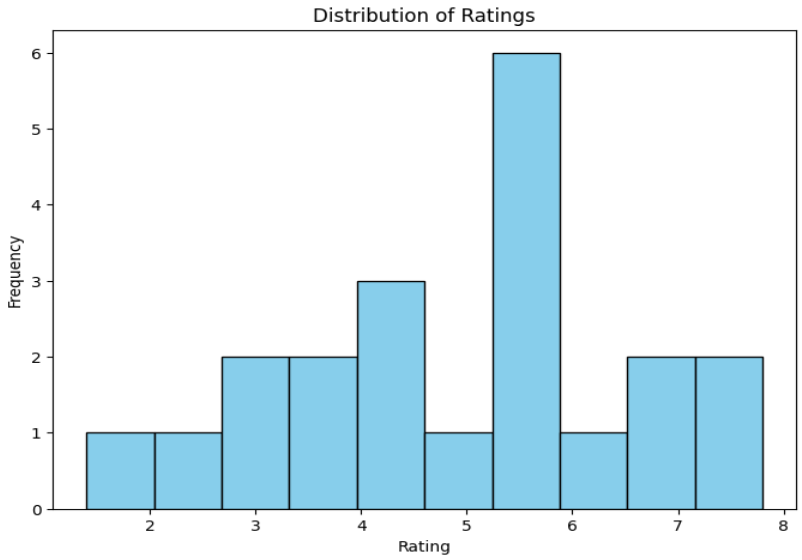
Question: Analyze the Easy Apply column. Does it contain boolean values or need transformation for better analysis?

In analyzing the Easy Apply column, it was observed that the column contained string values such as 'TRUE' and '-1' instead of boolean values. To address this, a replacement dictionary was created to map these string values to their corresponding boolean counterparts ('TRUE' to True and '-1' to False). The replace method was then used to replace the string values in the Easy Apply column with their boolean equivalents. Finally, the column was converted to boolean type to ensure consistency and better suitability for analysis. The resulting Easy Apply column now contains boolean values.

8. Rating Range:

Question: Investigate the range of values in the Rating column. Does it fall within expected rating scales, and how should outliers be treated?

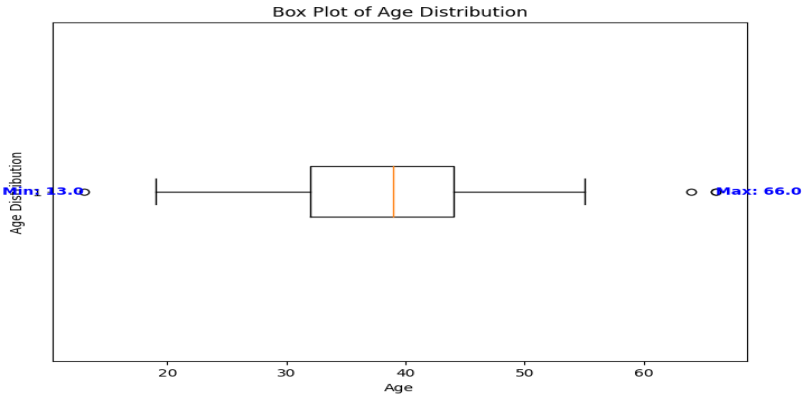
The code snippet addresses the investigation of the Rating column. Initially, it replaces values of -1, 0, None, and empty strings with NaN to handle missing or invalid entries. Then, it converts the 'Rating' column to numeric type, allowing for numerical analysis. To understand the distribution of ratings, the code generates a histogram that visualizes the frequency of different rating values. This visualization aids in assessing whether the range of values in the Rating column falls within expected rating scales and helps determine if outliers should be treated or not.



9. Age Distribution:

Question: Check the distribution of values in the Age column. Are there any unusual entries, and how might they impact analysis?

The code snippet first displays descriptive statistics (such as mean, median, min, and max) for the 'Age' column, providing an overview of the age distribution. It then creates a box plot visualizing the distribution of ages, with annotations for the minimum and maximum values. This analysis helps identify any unusual entries or outliers in the age data and assess their potential impact on further analysis.



10. Handling Special Characters:

Question: Examine all text-based columns (e.g., Location). Are there special characters or inconsistencies that need cleaning?

The provided code snippet examines text-based columns, such as 'Location', for special characters or inconsistencies that may require cleaning. It iterates through each cell in the specified column and checks for the presence of special characters defined in the **special_characters** list. If any cell contains such characters, it prints a message indicating the column name, index, and the cell value containing the special characters.

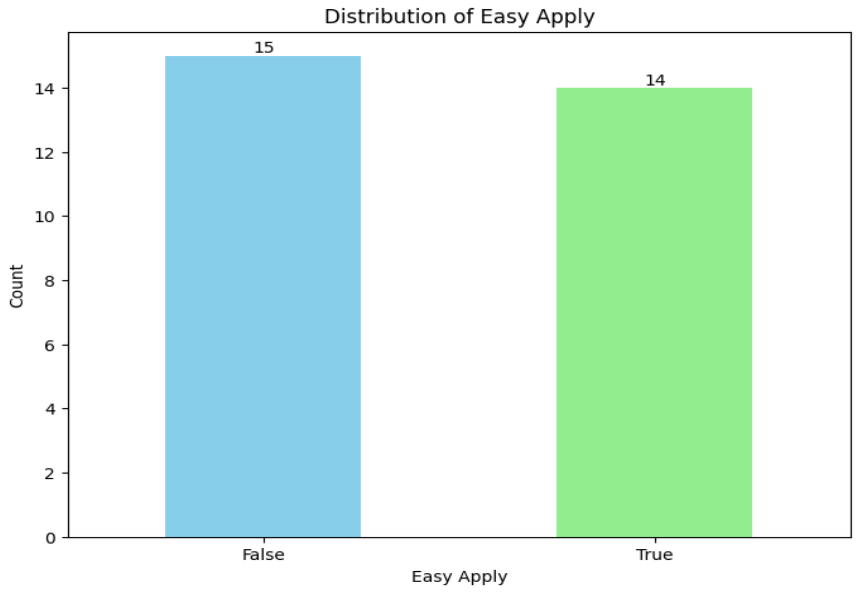
11. Data Integrity:

Question: Ensure data integrity by cross-referencing entries. For instance, does the Established column align with the Age column?

The provided code snippet checks for data integrity between the 'Established' and 'Age' columns in the DataFrame. It compares the calculated age of the company based on the established year with the value in the 'Age' column. If a discrepancy is found, it prints a message indicating a data integrity issue at the corresponding index. This process helps ensure the alignment of data entries and maintains data accuracy.

12. Easy Apply Transformation:

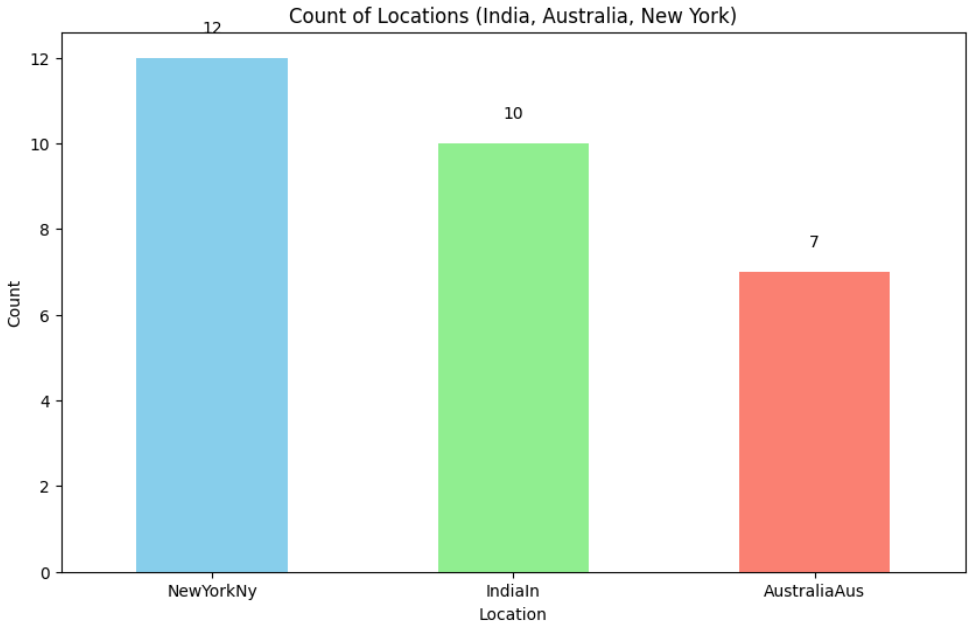
Question: If the Easy Apply column contains non-boolean values, how can it be transformed into a usable format?



13. Location Accuracy:

Question: Assess the accuracy of location entries. Are there misspelled or ambiguous locations that require correction?

To assess location accuracy, display unique values in the 'Location' column and manually inspect them for misspellings or inconsistencies. Make corrections as needed to ensure accurate and consistent location data.



14. Handling Categorical Data:

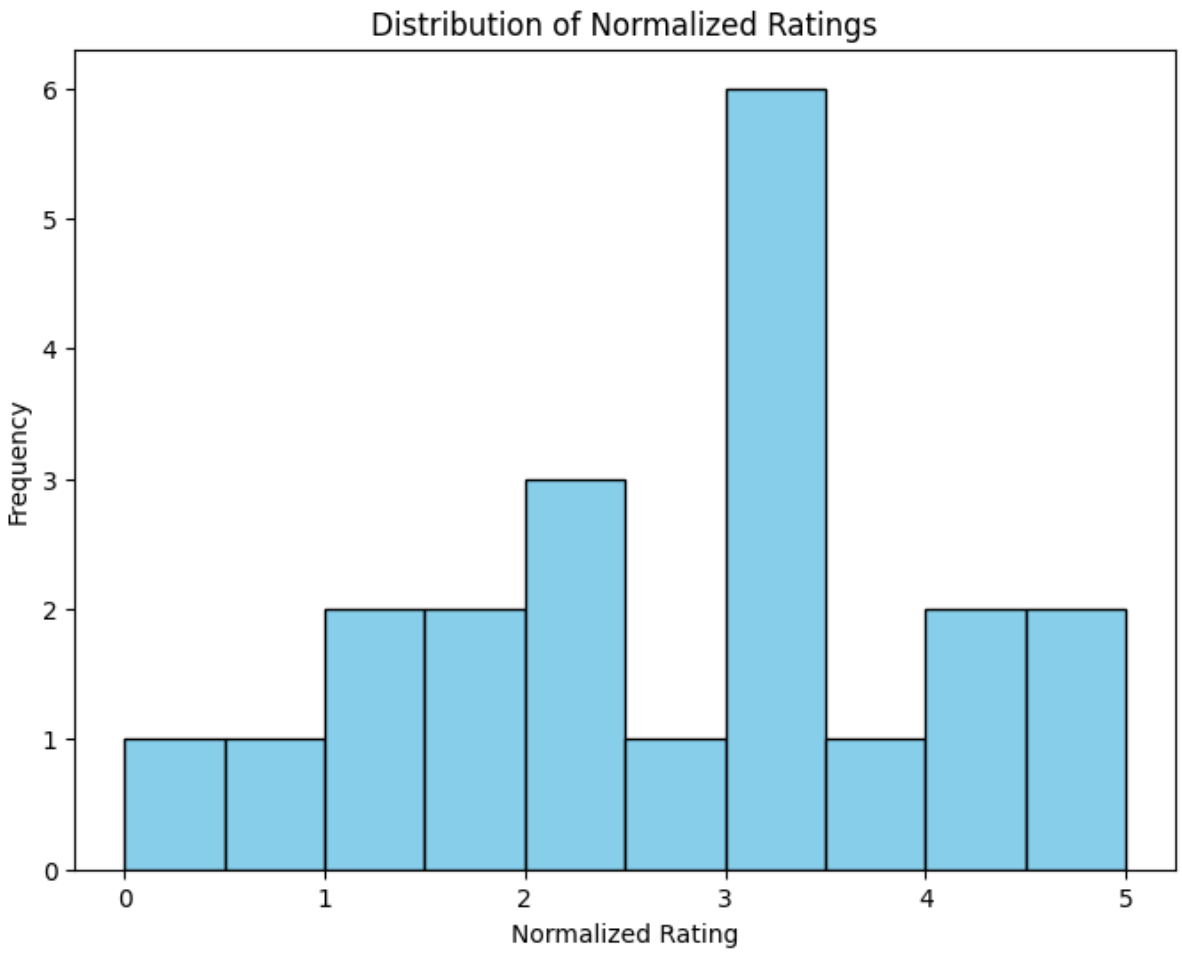
Question: For categorical indicators, consider encoding or transforming them into a format suitable for analysis?

To transform categorical indicators like the 'Location' column into a format suitable for analysis, one-hot encoding can be applied. This process converts categorical variables into a binary format, creating new binary columns for each category. In Python, this can be achieved using the get_dummies function from the pandas library. After encoding, the new columns can be concatenated with the original DataFrame. If necessary, the original categorical column can be dropped to avoid redundancy. The resulting DataFrame will contain the original data with the transformed categorical column suitable for analysis.

15. Consistent Rating Scale:

Question: Ensure a consistent rating scale in the Rating column. Should it be normalized or adjusted for uniform analysis?

To ensure a consistent rating scale in the 'Rating' column, the first step is to handle invalid values, such as -1, by replacing them with NaN (missing value) using the `replace` method. Once the invalid values are addressed, the ratings can be normalized to a consistent scale for uniform analysis. This can be achieved by scaling the ratings to a range of 0 to 5, where 0 represents the lowest rating and 5 represents the highest. The formula for normalization is ((x - min) / (max - min)) * 5, where x is the original rating, min is the minimum rating in the dataset, and max is the maximum rating in the dataset. After normalization, the transformed DataFrame can be examined to ensure that the rating scale is consistent and suitable for analysis.



By,

Rahulkannan