Documentation for Cleaning and Analyzing Indian Companies Data (2023 & 2024)

By Adnan Rasool

This documentation details the implementation of a Python script designed to clean, transform, and analyze datasets of the largest Indian companies for 2023 and 2024, sourced from Forbes.

Overview of the Script

The script achieves the following:

1. **Data Cleaning**:
    - Renames columns for simplicity.
    - Converts monetary values from ₹ crores to $ billions.
    - Cleans categorical data and standardizes percentages.
2. **Data Combination**:
    - Merges the datasets for 2023 and 2024 to enable year-over-year comparisons.
3. **Data Analysis**:
    - Computes changes in revenue, profit, and ranks of companies across years.
4. **Outputs**:
    - Cleaned individual datasets for 2023 and 2024.
    - A combined dataset for cross-year analysis.
    - A summary of year-over-year changes.

1. Helper Functions

`crore_to_billion_usd(value)`

- Converts monetary values from crores (₹) to billions ($), assuming **1 USD = 83 INR**.
- Handles strings, missing data, and invalid formats.

```
def crore_to_billion_usd(value):
    if pd.isna(value):
        return value
    if isinstance(value, str):
        value = value.replace('−', '-').replace(',', '')
        try:
            value = float(value)
        except ValueError:
            return np.nan
    return round((value / 100) / 83, 2)  # 1 crore = 10M INR
```

`clean_percentage(value)`

- Strips the `%` sign from strings and converts them to floats. Handles missing or malformed data.

```
def clean_percentage(value):
    if pd.isna(value):
```

```
        return value
    if isinstance(value, str):
        value = value.strip('%')
        try:
            return float(value)
        except ValueError:
            return np.nan
    return value
```

2. Data Cleaning

Cleaning the 2024 Dataset

The `clean_2024_data` function standardizes the 2024 data:

- Renames columns for better readability.
- Converts key columns (e.g., Revenue, Profit) to numeric types.
- Strips whitespace from categorical columns.

```
def clean_2024_data(df):
    df = df.rename(columns={
        'Revenue(billions US$)': 'Revenue',
        'Profit(billions US$)': 'Profit',
        'Assets(billions US$)': 'Assets',
        'Value(billions US$)': 'Value',
        'Forbes 2000 rank': 'Forbes_Rank'
    })
    numeric_columns = ['Revenue', 'Profit', 'Assets', 'Value']
    for col in numeric_columns:
        df[col] = pd.to_numeric(df[col], errors='coerce')
    df['Headquarters'] = df['Headquarters'].str.strip()
    df['Industry'] = df['Industry'].str.strip()
    df['Year'] = 2024
    return df
```

Cleaning the 2023 Dataset

The `clean_2023_data` function handles unique attributes in the 2023 dataset:

- Converts revenue and profit from ₹ crores to $ billions.
- Cleans growth percentages.
- Adds a `State_Controlled` indicator.

```
def clean_2023_data(df):
    df['Revenue'] = df['Revenue(in  ₹ Crore)'].apply(crore_to_billion_usd)
    df['Profit'] = df['Profits(in  ₹ Crore)'].apply(crore_to_billion_usd)
    df['Revenue_Growth'] = df['Revenue growth'].apply(clean_percentage)
    df['Headquarters'] = df['Headquarters'].str.strip()
    df['Industry'] = df['Industry'].str.strip()
    df['Year'] = 2023
    df['State_Controlled'] = df['State Controlled'].fillna('No')
```

```
    df['State_Controlled'] = df['State_Controlled'].map({'Yes': True, 'No': False})
    columns = ['Rank', 'Name', 'Industry', 'Revenue', 'Profit', 'Revenue_Growth',
               'Headquarters', 'State_Controlled', 'Year']
    return df[columns]
```

3. Data Combination

The `prepare_for_comparison` function merges the cleaned datasets, focusing on common columns.

```
def prepare_for_comparison(df_2024, df_2023):
    common_columns = ['Rank', 'Name', 'Industry', 'Revenue', 'Profit', 'Headquarters', 'Year']
    df_2024_comp = df_2024[common_columns].copy()
    df_2023_comp = df_2023[common_columns].copy()
    combined_df = pd.concat([df_2024_comp, df_2023_comp])
    return combined_df
```

4. Year-over-Year Analysis

The `get_year_over_year_changes` function calculates changes in revenue, profit, and rank between 2023 and 2024.

```
def get_year_over_year_changes(combined_df):
    changes = combined_df.pivot(index='Name', columns='Year', values=['Revenue', 'Profit', 'Rank'])
    changes['Revenue_Change'] = changes[('Revenue', 2024)] - changes[('Revenue', 2023)]
    changes['Profit_Change'] = changes[('Profit', 2024)] - changes[('Profit', 2023)]
    changes['Rank_Change'] = changes[('Rank', 2023)] - changes[('Rank', 2024)]
    return changes.sort_values('Revenue_Change', ascending=False)
```

5. Outputs

- Cleaned datasets:
  - `cleaned_india_companies_2024.csv`
  - `cleaned_india_companies_2023.csv`
- Combined dataset:
  - `combined_india_companies_2023_2024.csv` https://drive.google.com/file/d/1ibxXIM8sDFKfyU6biWfPdLXENuxmQgXj/view?usp=sharing
- Year-over-year changes summary (printed to console).

Sample Analysis Results

After running the script:

- Year-over-year changes in revenue, profit, and rank are calculated for each company.
- Example output (top 5 companies with the largest revenue increase):

Name	Revenue_Change	Profit_Change	Rank_Change
Reliance Industries	+5.0	+1.2	-2
TCS	+4.2	+0.8	-1
Infosys	+3.8	+1.0	0
HDFC Bank	+3.5	+1.5	+1

ICICI Bank	+3.0	+1.1	-3

links for dataset:
https://drive.google.com/file/d/1QKVvCmt-dFlkB93fClFk0G4JDWH-y4DA/view?usp=sharing

combined: https://drive.google.com/file/d/1ibxXIM8sDFKfyU6biWfPdLXENuxmQgXj/view?usp=sharing