Data Processing and Visualization Pipeline

Ryan Schlenz

Github Repository:
https://github.com/RyanSchlenz/schlenz_test_code.git

Overview

Objective:

Demonstrate a Python-based pipeline to:

1. Load demographic data from an XML file.
2. Clean and validate the data.
3. Generate a summary report (JSON).
4. Create a visualization (bar chart).

Workflow

Steps in the Pipeline:

1. Load raw XML data.
2. Clean the data (calculate ages, remove invalid records).
3. Generate a summary report of adults and children by city.
4. Visualize insights with a bar chart (average age by city).

Workflow Diagram:

- Input XML → Clean Data → JSON Report → Chart Visualization

Loading Data

Key Function: load_data

- **Input**: XML file with <person> elements.
- **Process**:
  - Parse XML with xml.etree.ElementTree.
  - Extract fields: id, name, dob, city, country.
  - Log invalid records to discarded_files.txt.
- **Output**: Pandas DataFrame.

Visual: [XML snippet and DataFrame example]

id	name	dob	city	country
7966824611	John Doe	1989-06-15	Springfield	USA
8699099618	Sally Doe	2010-04-03	Springfield	USA
8699099618	Mario Rodríguez	2000-05-03	SLC	USA
8699099618	Margaret Doe	19990101	Springfield	USA

```xml
<people>
    <person>
        <id>7966824611</id>
        <name>John Doe</name>
        <dob>1989-06-15</dob>
        <address>
            <street>123 Main St</street>
            <city>Springfield</city>
            <state>IL</state>
            <zipcode>62701</zipcode>
            <country>USA</country>
        </address>
    </person>
    <person>
        <id>8699099618</id>
        <name>Sally Doe</name>
        <dob>2010-04-03</dob>
        <address>
            <street>564 State St</street>
            <city>Springfield</city>
            <state>IL</state>
            <zipcode>77664</zipcode>
            <country>USA</country>
        </address>
    </person>
</people>
```

Cleaning Data

Key Function: clean_data

- **Process**:
  1. Calculate age from dob using calculate_age.
  2. Remove rows with missing values.
  3. Convert age column to integer.
- **Output**: Cleaned DataFrame, ready for analysis.

Visual:

Raw Data Example:

id	name	dob	city	country
7966824611	John Doe	1989-06-15	Springfield	USA
8699099618	Sally Doe	2010-04-03	Springfield	USA
8699099618	Mario Rodríguez	2000-05-03	SLC	USA
8699099618	Margaret Doe	19990101	Springfield	USA
6240350649	Zack Black	1930-03-01	SLC	[MISSING]

Cleaned Data Example:

id	name	dob	city	country	age
7966824611	John Doe	1989-06-15	Springfield	USA	34
8699099618	Sally Doe	2010-04-03	Springfield	USA	13
8699099618	Mario Rodríguez	2000-05-03	SLC	USA	23
8699099618	Margaret Doe	1999-01-01	Springfield	USA	24

Generating Report

Key Function: generate_report

- **Process**:
  1. Group data by city.
  2. Count adults (age > 18) and children (age <= 18).
  3. Save results as a JSON file.
- **Output**: JSON report summarizing population by city.

Visual:

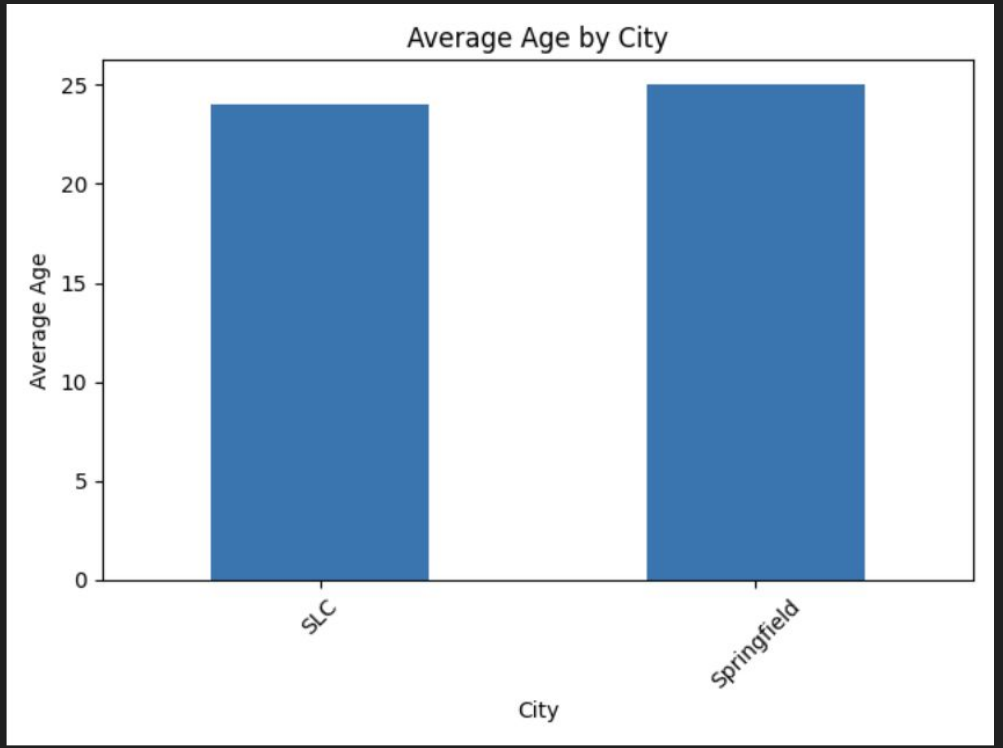

```
{
    "Springfield": {
        "adults": 2,
        "children": 1
    },
    "SLC": {
        "adults": 2,
        "children": 0
    }
}
```

Creating Visualization

Key Function: generate_chart

- **Process**:
  1. Calculate average age by city.
  2. Create a bar chart using matplotlib.
  3. Save the chart as an image file.
- **Output**: Bar chart (“Average Age by City”).

Visual:



Robustness and Scalability

Key Features:

- **Error Handling**:
  - Logs invalid records for debugging.
- **Modularity**:
  - Each function handles a specific task.
  - Easy to update or extend.
- **Scalability**:
  - Can handle larger datasets or additional data fields.
  - Adaptable for other data formats (e.g., JSON, CSV).

Conclusion

Key Takeaways:

- The pipeline processes raw data into actionable insights.
- Outputs include structured reports (JSON) and visualizations (charts).
- Robust, modular, and scalable design.