data-report

November 26, 2024

1 A Comparative Analysis of Solar Energy Infrastructure and Electric Vehicle Adoption Patterns

1.1 Data Sources

1.1.1 Solar Footprints Dataset (Dataset 1)

Why chosen : This dataset provides geospatial data on solar-powered electric generation facilities and related infrastructure in California, offering insights into renewable energy distribution in the state.

Source : https://opendatacommons.org/licenses/odbl/1-0/.

Data URL : https://cecgis-caenergy.opendata.arcgis.com/api/download/v1/items/9398e39a0424434b9e95ccf8e89

Why allowed to use : Openly available for public use as specified by the California Energy Commission

Obligations :

1. Attribute the California Energy Commission as the data source.
2. Please do not use the dataset for commercial purposes without explicit permission.

Content: Polygons representing the spatial footprints of solar energy infrastructure in California, derived from imagery interpretation and digitized polygons.

1.1.2 Electric Vehicle Population Data (Dataset 2)

Why chosen : This dataset tracks registered Battery Electric Vehicles (BEVs) and Plug-in Hybrid Electric Vehicles (PHEVs) in Washington, highlighting adoption patterns of clean energy consumption.

Source : http://opendatacommons.org/licenses/odbl/1.0/

Data URL : https://data.wa.gov/api/views/f6w7-q2d2/rows.csv?accessType=DOWNLOAD

Why allowed to use : The dataset is published under a standard open-data license for public access and research purposes.

Obligations :

1. Attribute the Washington State Department of Licensing.
2. Ensure data usage aligns with non-commercial and ethical research practices.

Content : It includes information on the number and types of electric vehicles registered in the state.

1.2 Data Pipeline Documentation

1.2.1 High-Level Overview

The data pipeline is designed to automate the process of downloading, cleaning, transforming, and saving data for analysis. It consists of the following steps:

1. **Data Downloading** : The pipeline fetches raw datasets from external sources (e.g., APIs or public repositories) and stores them locally.
2. **Data Cleaning** : Irrelevant columns are dropped, missing values are handled, and data formatting issues are fixed to ensure consistency.
3. **Data Transformation**: Data is transformed to align with the required schema, such as renaming columns, converting data types, and standardizing formats.
4. **Data Saving** : The cleaned data is saved in two formats:

a. SQLite databases for structured querying. **b**. Excel files for easy sharing and reporting.

1.2.2 Technologies Used

Programming Language : Python

Data Manipulation : Pandas

Database Management : SQLite

File Handling : Excel via openpyxl

Automation : Python's os and shutil modules for file management.

1.3 Data Cleaning and Transformation

Step	**Dataset**	**Description**
Remove Missing Values	Solar & Electric	Dropped rows with missing critical values to maintain consistency and completeness.
Standardize Columns	Solar & Electric	Renamed columns to snake_case to avoid errors in database queries.
Format Corrections	Electric Vehicle Dataset	Converted boolean values (1/0) to human-readable labels (Yes/No) for clarity.
Rate Transformation	Solar Dataset	Cleaned and standardized ratings from inconsistent formats (e.g., 4.5/5 to 4.5).

1.4 Problems and Solutions

Pipeline Challenges

1. **Irregular Formatting**: Columns like "rate" had inconsistent data formats (e.g., "4.5/5" and "N/A").

Solution: The regular expressions were used to extract and standardize numeric values.

2. **Duplicate Rows**: Several datasets contained duplicate entries.

Solution: I have implemented a deduplication step using Pandas' drop_duplicates method.

3. **Large Dataset Size**: Some datasets were too large to handle in memory.

Solution: I have processed the data in chunks using chunksize while reading and transforming.

1.5 Meta-Quality Measures and Error Handling

Aspect	**Description**
Validation Checks	- Ensured all column names adhered to snake_case naming conventions.
	- Verified that no critical columns contained missing or invalid data after cleaning.
Error Handling	- Used `try-except` blocks for robust error handling during file downloads and database operations. - Logged errors for debugging and flagged problematic records for manual review.
Handling Changing Input	- Dynamically adjusted schema mapping to handle new or missing columns in datasets. - Added flexibility in column renaming to prevent crashes due to unexpected schema changes.

1.6 Result and Limitation

1.6.1 Output Data

The output data is a cleaned, well-structured dataset in SQLite and Excel formats.

Data Structure : Relational tables stored in SQLite databases with consistent column names and types.

Data Quality: High, with no missing critical values or duplicate entries.

1.6.2 Output Format

SQLite : Chosen for its lightweight nature, suitability for structured querying, and ability to handle relational data efficiently.

Excel: Chosen for accessibility and ease of use in sharing and reporting.

1.6.3 Limitations

Input Dependency: The pipeline assumes a consistent schema in the input data. Drastic changes in the schema may require manual adjustments.

Large Dataset Scalability: While the current setup handles moderately large datasets, it may face performance issues with very large data volumes without additional optimization.

Boolean Conversions: Converting 1/0 values to “Yes/No” adds readability but could introduce issues if numeric analysis is required in subsequent steps.

1.7 Critical Reflection

1. The pipeline creates clean, organized data that’s ready for analysis.

2. Biases in the input data, like incomplete or uneven samples, might affect results.

3. Simplifying data for readability (like changing 1/0 to Yes/No) might make advanced analysis harder.

4. Regular checks of input data are needed to ensure it’s accurate and reliable.

5. The pipeline works well but depends on good-quality data from the start.