Lab 3 — Data Parsing and Scripting

Patrick F. Bloniasz

May 20, 2025

Total time: 4 h • Submit: GitHub repo URL on BlackBoard

Welcome to another day in lab

In this lab you will create a brand-new GitHub repository, set up a minimal Python environment from an `environment.txt` file, and write a short data-cleaning script in a Jupyter notebook. The goal is to see why messy real-world data needs fixing before analysis. This will be challenging, so consider working in groups and ask for help when stuck.

Recommended Repo Layout

```
your-lab-repo/          % top-level folder you create on GitHub
 data/
    messy_strings.csv  % we give you this file
 notebooks/
    clean_strings.ipynb % you create this today
 src/
    string_utils.py     % optional helper functions
 environment.txt         % we give you this file
 README.md               % create your own
```

0 Set up your repository from scratch

You'll build everything yourself.

1. Create a new repo on GitHub: `your-username/lab-3-data-parsing`.
   - Visibility: Public is fine, but Private is okay too, just at bloniaszp (pblonias@bu.edu) as a collaborator.
2. In VS Code → Terminal run:
   ```
   git clone https://github.com/lab-3-data-parsing.git
   cd lab-3-data-parsing
   ```
3. Download `environment.txt` and `messy_strings.csv` and place:

  - `environment.txt` at the repo root
  - `messy_strings.csv` inside `data/` (create `data/` if needed)

4. Commit starter files:

   ```
   git add .
   git commit -m "Add starter files (env and messy data)"
   git push
   ```

1 Create and activate the Python environment

We'll use the packages in `environment.txt`.

Inspect `environment.txt`:

```
pytest
numpy
matplotlib
scipy
ipykernel
notebook
```

Install (choose one):

- Using mamba/conda:

  ```
  mamba create -n lab_3 --file environment.txt -y
  mamba activate lab_3
  ```

Test:

```
python -c "import pandas, matplotlib; print(’all good!’)"
```

Note from Patrick: Commit only `environment.txt`—never commit a `venv/` folder if you hate doing things that work well, like mamba. If `venv` doesn't mean anythign to you right now, good, it's not worth using anymore.

2 Open Jupyter and start coding

In the VS Code browser go to `notebooks/`, create a New Notebook, save as `clean_strings.ipynb`.

Useful Functions

- `pd.read_csv`
- `Series.dropna`
- `Series.astype`
- `Series.str.strip`
- `Series.str.lower`

- `Series.str.replace`
- `string.punctuation`
- `re.escape`
- `Series[condition]`
- `Series.nunique`
- `Series.value_counts`
- `DataFrame.to_csv`
- `Series.plot.bar`
- `plt.show()`

What is a *Series*?

In **pandas**, a *Series* is a one-dimensional labeled array that can hold data of any type (integers, strings, floats, etc.). You can think of it like a single column in a spreadsheet:

- Each entry in the Series has an *index* (the row label) and a *value*.
- Series methods (like `.dropna()`, `.str.strip()`, or `.value_counts()`) operate element-wise across all values.
- When we write `df['raw']`, we are selecting the "`raw`" column of the DataFrame as a Series.

3 Lab Tasks

Use one cell per task.

1. **Task A:** Load `data/messy_strings.csv` into `df_raw`.
   Hint: `pd.read_csv()`
2. **Task B:** Write `clean_strings(strings)` that:
   - Strips spaces
   - Lower-cases
   - Removes punctuation (!?,.;:)
   - Drops empty entries
3. **Task C:** Apply it: `df['clean'] = clean_strings(df['raw'])`
4. **Task D:** Compute on `df['clean']`:
   - Total rows
   - Unique count
   - Most common string (`value_counts()`)

5. **Task E:** Save cleaned data:

   ```
   df.to_csv('data/messy_strings_clean.csv', index=False)
   ```

6. **Task F (Bonus):** Plot top-5 strings:

   ```
   df['clean'].value_counts().head(5).plot.bar()
   ```

7. **Task G:** Commit and push changes:

   ```
   git add -A && git commit -m "Finish lab tasks" && git push
   ```

4 Stretch Goals (optional)

- Move `clean_strings` into `src/string_utils.py` and write a unit test with `assert`
- Add `LICENSE` and `.gitignore`
- Use `string.punctuation` instead of hard-coding

5 How to hand in

- Ensure the notebook runs top-to-bottom without errors
- Push your final commit
- Paste your GitHub repo URL into the course submission form on BlackBoard.