Reproducible Research Workflows

Note on data. This assignment uses a **synthetic** voter-turnout dataset provided by `poliscitools` (via `example_data`). The goal is to practice reproducibility tooling (dependency management, logging, packaging outputs, and replication checks)—not to draw substantive inferences about real voters.

Conceptual Questions

Please write three to ten sentence explanations for each of the following questions. **You are only required to answer ONE of the two questions below.**

1. Explain what problem `renv` solves in reproducible research. In your answer, describe what information is stored in `renv.lock`, what `renv::restore()` does, and why sharing code without dependency versions can fail replication even when the analysis is "correct."
2. Explain why logging (e.g., `logger`) is part of professional, reproducible analysis. Give two concrete examples of what you would log in a pipeline (inputs, parameters, random seeds, file paths, model summaries, warnings), and explain how logs help diagnose non-reproducible results.

Applied Exercises

Use the code in the week's code tutorial and the lecture slides to answer the following questions.

Deliverables checklist (what your repo must contain):

- `renv.lock` (committed),
- your analysis script(s) (e.g., in `analysis/` or `scripts/`),
- `outputs/figures/` containing your plot(s),
- a log file (e.g., `analysis_log.txt`) that records what ran and what files were written,
- a short reproducibility note (e.g., `REPRODUCE.md`) describing how to re-run your analysis from a fresh clone.

3. **Clone the instructor reproducibility folder and start your project.**
   - Clone the instructor workflow repository and work inside the `reproducibility/` folder (only that folder is relevant for this assignment).
   - Create your own GitHub repository for submission and push your work there.
   - Evidence (include screenshots or paste into `commands.log`):
     - `git clone ...`
     - `cd reproducibility`
     - `git status`
   - Helpful tidbit:

   - macOS Terminal / Windows PowerShell both support `pwd`, `ls`, and `cd`.
   - If `git` is "not recognized", install Git and restart your terminal.
4. **Reproducible workflow + three regressions + plot (Income as DV).**
   - **Project structure:** Ensure your folder includes (at minimum)
     - `data/raw/` (store the attached CSV here),
     - `data/processed/` (optional, only if you create derived data),
     - `outputs/figures/` and `outputs/tables/`,
     - `analysis_log.txt`.
   - **Dependency capture with `renv`:**
     - Initialize or restore `renv` in your project.
     - Create and commit `renv.lock` (use `renv::snapshot()` once your code runs).
   - **Run three different regressions (hard-coded, sequential) using `income` as the dependent variable:**
   - Save a simple regression table (or clearly printed model summaries) to `outputs/tables/`.
   - **Logging requirement:** Create/update `analysis_log.txt` so it records:
     - when the analysis was run,
     - the number of rows loaded,
     - which outputs were written (filenames/paths),
     - the output of `sessionInfo()` written to a file (e.g., `outputs/session_info.txt`).
5. **Push your changes to your repository (submission).**
   - Commit and push:
     - your analysis script(s),
     - `renv.lock`,
     - `analysis_log.txt` and `REPRODUCE.md`,
     - your output plot(s) and any small tables.
   - Do *not* commit large intermediate files.
   - Submission: submit the link to **your** GitHub repository.
6. **Bonus (if you finish early): Bootstrap + perfect replicability.**
   - Re-run the analysis using bootstrap simulations (e.g., resample rows and re-fit at least one of your models many times).
   - Set a seed **once** at the top (use `set.seed(123)`).
   - Save bootstrap results to `outputs/tables/` (e.g., coefficient distributions / intervals).
   - Run the full analysis at least **twice** from a fresh R session and verify the bootstrap outputs are **identical**.
   - In 4–6 sentences, explain what threatened replicability and what you did to eliminate it.