In-Class Problem Set: Distributions and Overplotting with Flight Data (R + GitHub)

Goal. Practice visualizing and interpreting distributions using real transportation data. You will use histograms and density plots to explore skewed variables, compare groups, and diagnose overplotting. You will submit your work through GitHub using a reproducible workflow.

Dataset. This problem set uses the `nycflights13` dataset, which contains detailed information on flights departing from NYC airports.

Key variables.

- `dep_delay`: Departure delay (minutes)
- `arr_delay`: Arrival delay (minutes)
- `air_time`: Time in the air (minutes)
- `distance`: Flight distance (miles)
- `carrier`: Airline carrier code
- `origin`: Airport of origin (JFK, LGA, EWR)

What to submit (in your GitHub repo).

- A script file: `scripts/lab.R`
- A short write-up: `outputs/writeup.md`
- Saved figures in `figures/` (see requirements below)

Rules.

- Work inside an **R Project**.
- Use a **sequential, hard-coded workflow** (no user-defined functions).
- Save all plots using `ggsave()` (no screenshots).
- Git commands must be run in the **Terminal tab**, not the R Console.

Questions

Pseudo-code expectation. For Questions 2–5, include short pseudo-code comments in `scripts/lab.R` before each major block (for example: "load data → filter missing values → plot → save figure").

1. **Pull the data and set up your workflow (proof required).**
   (a) Pull the latest course repository from GitHub:

   ```
   git status
   git pull
   ```

   (b) Confirm you can load the `nycflights13` package in R.
   (c) **Proof (write-up):** In `outputs/writeup.md`, paste:
   - the output of `getwd()`,
   - the output of `head(nycflights13::flights)`.

2. **Explore skew with histograms.**
   Create histograms for the following variables:
   - `dep_delay`
   - `arr_delay`

   Required:
   - Start this section in your script with 3–5 lines of pseudo-code that outline your workflow.
   - Use at least **two different bin widths** for each variable.
   - Decide whether to include or exclude extreme values and justify your choice.

   Save your figures as:
   - `figures/dep_delay_hist.png`
   - `figures/arr_delay_hist.png`
3. **Histogram vs density: same data, different views.**
   For `dep_delay`, create:
   - one histogram, and
   - one density plot.

   Suggested edit:
   - Add 2–4 lines of pseudo-code before writing code for this question.
   - Use the same x-axis limits so the plots are comparable.
   - Make clear in labels what each plot represents.

   Save your figures as:
   - `figures/dep_delay_hist_vs_density.png`
4. **Grouped distributions.**
   Compare the distribution of `dep_delay` across:
   - **origins** (JFK, LGA, EWR), or
   - **carriers** (choose at least three major carriers).

   Create one plot that shows all groups together using color or faceting.
   Important: You must address overplotting explicitly (e.g., overlapping densities or stacked histograms).
   Pseudo-code prompt: Write 2–4 pseudo-code lines describing your grouping strategy and whether you will facet or map color.
   Save your figure as:
   `figures/grouped_dep_delay.png`
5. **Overplotting and transparency.**
   Create a scatter plot with:
   - x-axis: `distance`
   - y-axis: `air_time`

   Then:
   - Add 2–3 pseudo-code lines for your scatterplot plan (raw plot → alpha plot → compare).
   - produce one version without transparency,
   - produce a second version using transparency (`alpha`).

   Save your figures as:
   - `figures/airtime_distance_raw.png`
   - `figures/airtime_distance_alpha.png`
6. **Interpretation (write-up required).**
   In `outputs/writeup.md`, write 12–16 sentences addressing:
   - How skew appears in the delay variables.
   - How bin width changes what patterns are visible.

- What information density plots emphasize relative to histograms.
- How grouping changes your interpretation of delays.
- Why transparency matters in the scatter plot.

7. **Submit your work (GitHub or Canvas; proof required).**
   (a) Choose **one** submission path:
   - **GitHub path:** In the **Terminal tab**, run:

     ```
     git status
     git add .
     git commit -m "Distribution and overplotting lab: flights data"
     git push
     ```

   - **Canvas path:** Upload `scripts/lab.R`, `outputs/writeup.md`, and required files from `figures/` to Canvas.

   (b) **Proof (write-up):** Paste:
   - if using GitHub: the output of `git status` after committing and `git log -1`,
   - if using Canvas: a short note confirming upload date/time and the list of uploaded files.

Optional challenge (if you finish early)

Choose one distribution (e.g., `dep_delay`) and create a version that improves interpretability for a **general audience**. In 5–7 sentences, explain:

- what design choices you changed (bins, scale, labels, grouping),
- what you simplified or emphasized,
- and why these choices help a non-technical viewer.

Checklist (before you leave)

- `scripts/lab.R` runs top-to-bottom
- Required figures exist in `figures/`
- `outputs/writeup.md` includes interpretation + proofs
- Work is submitted (either pushed to GitHub or uploaded to Canvas)