Exploratory Data Analysis in R with Tidyverse

Session 4: Functional Programming

instats

Functional Programming for EDA

- Use `map()` and `walk()` to apply functions across columns, lists, or features
- Automate repetitive summaries or visualizations with minimal code
- Keep your analysis tidy and readable by avoiding loops
- Combine `group_split()` with `map_*()` for per-group operations

Automating Feature Summaries and Plots

- Define a single function that handles both data summaries and plots
- Use `map()` to loop across variables like energy, tempo, or valence
- Combine summary stats with histograms or violin plots
- Add consistent labeling, styling, and export settings

Extracting Patterns Across Variables

- Use `imap_lgl()` to search all character columns at once
- Build quick “search engines” for exploratory keyword filtering
- Detect presence of terms like "love" or "may" across multiple fields
- Quickly identify which variables hold the information you're after

Nesting, Mapping, and Combining Groups

- Use `group_split()` to split a dataset into a list by group
- Apply `map()` to summarize or analyze each group separately
- Combine results using `reduce()` or `bind_rows()`
- Perfect for by-category EDA, such as genre-level summaries

Using `walk()` for Action-Oriented EDA

- Use `walk()` to apply a function with side effects (like saving files)
- Automate plots for multiple groups with consistent formatting
- Avoid copy-pasting `ggplot()` code by passing each group dynamically
- Helpful for building reports or exporting batch outputs