Regular Expressions in R (Functions & Validation)

Jared Edgerton

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. In a data science workflow, why is it useful to wrap data-cleaning logic into functions (rather than writing one-off code blocks)? In your answer, connect your explanation to regex-based parsing/validation and mention one way you would test that your function behaves correctly.
2. When writing regex for data extraction, what is the difference between (i) identifying a match, (ii) capturing components of a match, and (iii) replacing a match? Use a concrete example from either phone-number standardization or password validation to illustrate your answer.

Applied Exercises

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

3. **Phone Number Formatting:** Create a regex pattern to identify and format phone numbers in text. Your function should work for different formats:
   - `(123) 456-7890`
   - `123-456-7890`
   - `1234567890`

   and standardize them to: `(XXX) XXX-XXXX`.

   Requirements:
   - Implement `format_phone_numbers(text)` using `stringr` (e.g., `str_replace_all()`).
   - Demonstrate that it works on a character vector with **at least 6** test strings, including at least one string that contains **two** phone numbers.
   - Show the **before** and **after** output clearly (print a small table is fine).

   Starter skeleton (you may copy/paste):

   ```
   # TODO: Implement this function.
   format_phone_numbers <- function(text) {
     # pattern <- ...
     # Use str_replace_all / str_replace to standardize
   }
   ```

4. **Password Strength Checker:** Implement a function that uses regex to check the strength of a password.

   Criteria:

- At least 8 characters long
- At least one uppercase letter
- At least one lowercase letter
- At least one digit
- At least one special character (`@$!%*?&`)

Requirements:

- Implement `check_password_strength(password)` using `stringr` (e.g., multiple `str_detect()` checks, or a single combined pattern).
- Create a test vector with **at least 8** passwords (mix of valid/invalid).
- Output a small results table with columns like `password` and `is_strong` (and optionally a column indicating what criterion failed).

Starter skeleton (you may copy/paste):

```
# TODO: Implement this function.
check_password_strength <- function(password) {
  # Use str_detect with multiple patterns (or a single combined pattern)
}
```

5. **Challenge Question (Optional — if you finish early):** Write a minimal set of unit tests for your two functions using `testthat`.
   - Create at least **5** tests for `format_phone_numbers()` and at least **5** tests for `check_password_strength()`
   - Include at least one edge case for each (e.g., numbers embedded in longer digit strings; passwords that fail exactly one rule).
   - Show that your tests run without failing.