Principles of webscraping

Etienne Bacher

LISER

2022-08-04

The three laws of ~~robotics~~ webscraping

The three laws of webscraping

1. Don't be in a hurry
2. Separate the scraping from the formatting
3. Anticipate failure

1. Don't be in hurry

The scraping shouldn't be too fast for two main reasons:

1. For the website owner: don't flood the server with requests (not all websites you want to scrape are owned by a big company);
2. For you: don't get kicked out / banned.

2. Separate the scraping from the formatting

Scraping takes time (cf previous slide).

You should minimize the risk of crashing and losing your progress, which means minimizing the operations at each iteration.

We have:

1. lots of storing space on our laptops
2. not a lot of time

Save the raw HTML in external files first. You will be able to clean it later (and offline).

3. Anticipate failure

Minimizing the risk of failures doesn't mean that there won't be any failures.

Your code should be resilient to these failures and keep running.

In R, this can be done with `try()` and `tryCatch()`.

3. Anticipate failure

Let's create a dummy function that takes `x` and:

- returns `x*2` if `x >= 0`
- errors if `x < 0`

```r
dummy_fun <- function(x) {
  if (x < 0) stop("x is not big enough")
  return(x*2)
}
dummy_fun(-1)
```

```
## Error in dummy_fun(-1): x is not big enough
```

```r
dummy_fun(1)
```

```
## [1] 2
```

3. Anticipate failure

What happens if we loop through this function from -1 to 1?

```
for (i in -1:1) {
  print(i)
  dummy_fun(i)
}
```

```
## [1] -1
```

```
## Error in dummy_fun(i): x is not big enough
```

Because the function doesn't handle errors, *the error stops the whole loop* (we never run `dummy_fun(0)` and `dummy_fun(1)`).

3. Anticipate failure

```r
x <- try(dummy_fun(-1), silent = TRUE)
class(x)
```

```
## [1] "try-error"
```

```r
for (i in -1:1) {
  print(i)
  x <- try(dummy_fun(i), silent = TRUE)

  if (inherits(x, "try-error")) {
    NULL
  } else {
    x
  }
}
```

```
## [1] -1
## [1] 0
## [1] 1
```

Note: you could also use `try()` directly in `dummy_fun()`.

3. Anticipate failure

```r
x <- tryCatch(
  dummy_fun(-1),
  error = function(e) {
    print("This is a custom error message")
    print(paste("Original error:", e))
  }
)
```

```
## [1] "This is a custom error message"
## [1] "Original error: Error in dummy_fun(-1): x is not big enough\n"
```

```r
class(x)
```

```
## [1] "character"
```