UW Datathon 2026 Preprocessing

Kirby Vo, Kelbin Luong

2026-02-07

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.4     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```
library(stringr)
library(writexl)
```

LOAD DATA

```
data <- read_delim("datasets/Access_to_Tech_Dataset.csv")
```

```
## Rows: 3524 Columns: 17
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (14): id, domain_category, web_URL, scrape_status, html_file_name, html_...
## dbl  (3): web_URL_id, violation_count, violation_score
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

PREVIEW DATA

```
data |>
  names()
```

```
##  [1] "id"                          "web_URL_id"
##  [3] "domain_category"             "web_URL"
##  [5] "scrape_status"               "html_file_name"
##  [7] "html_file_path"              "violation_count"
##  [9] "violation_name"              "violation_score"
## [11] "violation_description"       "violation_description_url"
## [13] "affected_html_elements"      "violation_category"
## [15] "violation_impact"            "wcag_reference"
## [17] "supplementary_information"
```

```
data |>
  sample_n(10)
```

```
## # A tibble: 10 x 17
##    id            web_URL_id domain_category  web_URL scrape_status html_file_name
##    <chr>              <dbl> <chr>            <chr>   <chr>         <chr>
##  1 6699878654_4 6699878654 News and Media   https:~ scraped       www_propublic~
##  2 5117799549_8 5117799549 E-commerce       https:~ scraped       www_alibaba_c~
##  3 193670859_12  193670859 News and Media   https:~ scraped       www_abcnews_g~
##  4 1851235277_1 1851235277 E-commerce       https:~ scraped       www_rakuten_c~
##  5 501_1               501 Health and Well~ https:~ scraped       www_healthcar~
##  6 7124602025_2 7124602025 News and Media   https:~ scraped       www_bostonglo~
##  7 8056970126_5 8056970126 Streaming Platf~ https:~ scraped       www_grokker_c~
##  8 1488388154_2 1488388154 Technology Scie~ https:~ scraped       www_cnet_com_~
##  9 503_6               503 Technology Scie~ https:~ scraped       arstechnica_c~
## 10 8866935042_2 8866935042 Streaming Platf~ https:~ scraped       www_fox_com_h~
## # i 11 more variables: html_file_path <chr>, violation_count <dbl>,
## #   violation_name <chr>, violation_score <dbl>, violation_description <chr>,
## #   violation_description_url <chr>, affected_html_elements <chr>,
## #   violation_category <chr>, violation_impact <chr>, wcag_reference <chr>,
## #   supplementary_information <chr>
```

SEARCH FOR DATA INCONSISTENCIES

We found that looking at distinct categories, we saw some inconsistent domain category entries that we would need to remap

```
data |>
  distinct(domain_category)
```

```
## # A tibble: 9 x 1
##   domain_category
##   <chr>
## 1 Government and Public Services
## 2 News and Media
## 3 Technology Science and Research
## 4 E-commerce
## 5 Educational Platforms
## 6 Streaming Platforms
## 7 Health and Wellness
## 8 TechnologyScienceResearch
## 9 Ecommerce
```

We also found cases where if a url had multiple domain categories assigned to it, any violation entries for it would be duplicated accordingly.

```
data |>
  filter(web_URL == "https://arstechnica.com/health/") |>
  select(domain_category, web_URL, violation_name, violation_score) |>
  head(2)
```

```
## # A tibble: 2 x 4
##   domain_category                  web_URL         violation_name violation_score
##   <chr>                            <chr>           <chr>                    <dbl>
## 1 News and Media                   https://arstec~ color-contras~               4
## 2 Technology Science and Research https://arstec~ color-contras~               4
```

We had also noticed while processing the data that there were instances of urls with or without /, which would count them as seperate webpages despite being the same, which needed to be addressed.

```
data |>
  filter(web_URL == "https://www.nbcnews.com" | web_URL == "https://www.nbcnews.com/") |>
  select(web_URL) |>
  slice(1, 22)
```

```
## # A tibble: 2 x 1
##   web_URL
##   <chr>
## 1 https://www.nbcnews.com
## 2 https://www.nbcnews.com/
```

Regarding domains, we found that there were a lot of domains such as arstechnica.com which had multiple webpages under it, which meant we would need to group them properly by domain in order to answer domain specific questions.

```
data |>
  filter(str_detect(web_URL, "arstechnica\\.com")) |>
  distinct(web_URL) |>
  head(5)
```

```
## # A tibble: 5 x 1
##   web_URL
##   <chr>
## 1 https://arstechnica.com/ai/
## 2 https://arstechnica.com/information-technology/
## 3 https://arstechnica.com/cars/
## 4 https://arstechnica.com/culture/
## 5 https://arstechnica.com/gaming/
```

CLEAN DATA

Here we create a map to convert weird domain category values to correct ones.

```r
renaming_map <- c(
  "Government and Public Services" = "Government and Public Services",
  "News and Media" = "News and Media",
  "Technology Science and Research" = "Technology Science and Research",
  "E-commerce" = "E-commerce",
  "Educational Platforms" = "Educational Platforms",
  "Streaming Platforms" = "Streaming Platforms",
  "Health and Wellness" = "Health and Wellness",
  "TechnologyScienceResearch" = "Technology Science and Research",
  "Ecommerce" = "E-commerce"
)

data_clean <- data |>
  mutate(
    domain_category = renaming_map[domain_category],
    domain = web_URL |>
      str_replace("^https?://(www\\.)?", "") |>
      str_replace("/.*$", "")
  )
```

Ensure that violations aren't duplicated based off numebr of additional domain categories.

```r
data_clean_deduped <- data_clean |>
  distinct(web_URL,
           violation_name,
           violation_score,
           violation_category,
           violation_impact,
           wcag_reference,
           .keep_all = TRUE)
```

VERIFY PROBLEMS FIXXED

```r
data_clean_deduped |>
  distinct(domain_category)
```

```
## # A tibble: 7 x 1
##   domain_category
##   <chr>
## 1 Government and Public Services
## 2 News and Media
## 3 E-commerce
## 4 Technology Science and Research
## 5 Educational Platforms
## 6 Streaming Platforms
## 7 Health and Wellness
```

```r
data_clean_deduped |>
  filter(web_URL == "https://www.nbcnews.com/" | web_URL == "https://www.nbcnews.com") |>
  select(web_URL, domain) |>
  slice(c(1, 22))
```

```
## # A tibble: 2 x 2
##   web_URL                  domain
##   <chr>                    <chr>
## 1 https://www.nbcnews.com  nbcnews.com
## 2 https://www.nbcnews.com/ nbcnews.com
```

```
data_clean_deduped |>
  filter(web_URL == "https://arstechnica.com/health/") |>
  select(domain_category, web_URL, violation_name, violation_score) |>
  head(2)
```

```
## # A tibble: 2 x 4
##   domain_category web_URL                         violation_name violation_score
##   <chr>           <chr>                           <chr>                    <dbl>
## 1 News and Media  https://arstechnica.com/health/ color-contras~               4
## 2 News and Media  https://arstechnica.com/health/ color-contrast               4
```

AGGREGATE DATA

Here we aggregated data per page for potentially useful metrics to include in individual summaries we will input into tableau and graph. (We didn't end up using the aggregated values calculated here.)

```
page_summary <- data_clean_deduped |>
  group_by(web_URL, domain) |>
  summarise(
    n_violations = n(),
    avg_severity = mean(violation_score, na.rm = TRUE),
    max_severity = max(violation_score, na.rm = TRUE),
    min_severity = min(violation_score, na.rm = TRUE),
    n_distinct_violation_types = n_distinct(violation_name),
    n_distinct_violation_categories = n_distinct(violation_category),
    .groups = "drop"
  )
```

DOMAIN-LEVEL SUMMARY

```
domain_summary <- page_summary |>
  group_by(domain) |>
  summarise(
    n_pages = n(),
    total_violations = sum(n_violations, na.rm = TRUE),
    avg_violations_per_page = mean(n_violations, na.rm = TRUE),
    avg_severity = mean(avg_severity, na.rm = TRUE),
    .groups = "drop"
  )
```

```
domain_summary |>
  head(4)
```

```
## # A tibble: 4 x 5
##   domain         n_pages total_violations avg_violations_per_page avg_severity
##   <chr>            <int>            <int>                   <dbl>        <dbl>
## 1 3dcart.com           1                4                       4         4.5
## 2 abc.net.au           1                4                       4         3.75
## 3 abcnews.go.com       1               13                      13         4.08
## 4 academia.edu         1                8                       8         3.5
```

```
domain_category_summary <- data_clean |>
  group_by(domain_category) |>
  summarise(
    total_violations = n(),
    avg_severity = mean(violation_score, na.rm = TRUE),
    .groups = "drop"
  )
```

```
domain_category_summary |>
  head(4)
```

```
## # A tibble: 4 x 3
##   domain_category                total_violations avg_severity
##   <chr>                                     <int>        <dbl>
## 1 E-commerce                                  345         3.56
## 2 Educational Platforms                       575         3.63
## 3 Government and Public Services              380         3.40
## 4 Health and Wellness                         165         3.54
```

VIOLATION-LEVEL SUMMARIES

```
violation_summary <- data_clean |>
  count(violation_name, sort = TRUE) |>
  rename(violations = n)

violation_type_by_domain_category <- data_clean |>
  group_by(domain_category, violation_name) |>
  summarise(violations = n(),
            .groups = "drop")

violation_category_by_domain_category <- data_clean |>
  filter(!is.na(violation_category)) |>
  group_by(domain_category, violation_category) |>
  summarise(violations = n(),
            .groups = "drop")


violation_category_summary <- data_clean |>
  filter(!is.na(violation_category)) |>
  count(violation_category, sort = TRUE) |>
  rename(violations = n)
```

```r
top_violations <- violation_summary |>
  slice_max(violations, n = 10) |>
  pull(violation_name)

violation_summary_grouped <- violation_summary |>
  mutate(
    violation_group = if_else(
      violation_name %in% top_violations,
      violation_name,
      "Other"
    )
  ) |>
  group_by(violation_group) |>
  summarise(violations = sum(violations), .groups = "drop")
```

```r
violation_summary  |>
  head(4)
```

```
## # A tibble: 4 x 2
##   violation_name          violations
##   <chr>                        <int>
## 1 color-contrast-enhanced        502
## 2 region                         382
## 3 color-contrast                 245
## 4 duplicate-id                   230
```

```r
violation_type_by_domain_category  |>
  head(4)
```

```
## # A tibble: 4 x 3
##   domain_category violation_name    violations
##   <chr>           <chr>                  <int>
## 1 E-commerce      aria-allowed-attr         12
## 2 E-commerce      aria-allowed-role          5
## 3 E-commerce      aria-command-name          2
## 4 E-commerce      aria-dialog-name           2
```

```r
violation_category_by_domain_category  |>
  head(4)
```

```
## # A tibble: 4 x 3
##   domain_category       violation_category violations
##   <chr>                 <chr>                   <int>
## 1 E-commerce            Layout                     73
## 2 E-commerce            Syntax                    272
## 3 Educational Platforms Layout                    159
## 4 Educational Platforms Semantic                   26
```

```r
violation_category_summary|>
  head(4)
```

```
## # A tibble: 3 x 2
##   violation_category violations
##   <chr>                   <int>
## 1 Syntax                   2634
## 2 Layout                    853
## 3 Semantic                   33
```

```
violation_summary_grouped  |>
  head(4)
```

```
## # A tibble: 4 x 2
##   violation_group         violations
##   <chr>                        <int>
## 1 Other                         1151
## 2 color-contrast                 245
## 3 color-contrast-enhanced        502
## 4 duplicate-id                   230
```

EXPORT EVERYTHING FOR TABLEAU

```
write.csv(domain_summary, "processed_data/domain_summary.csv", row.names = FALSE)
write.csv(domain_category_summary, "processed_data/domain_category_summary.csv", row.names = FALSE)
write.csv(violation_summary, "processed_data/violation_summary.csv", row.names = FALSE)
write.csv(violation_summary_grouped, "processed_data/violation_summary_grouped.csv", row.names = FALSE)
write.csv(violation_category_summary, "processed_data/violation_category_summary.csv", row.names = FALSE
write.csv(violation_type_by_domain_category, "processed_data/violation_type_by_domain.csv", row.names =
write.csv(violation_category_by_domain_category, "processed_data/violation_category_by_domain.csv", row
```

finished :D