Project title

Appendix to report

marvelous-echidna

Data cleaning

The raw data is read from a CSV file named 'nasa_global_landslide_catalog_point.csv' located in the 'data' directory using the read_csv() function from the readr package. The show_col_types = FALSE argument is used to suppress the display of column data types.

The select() function is used to keep only the relevant columns needed for the analysis. The selected columns are 'event_date', 'location_accuracy', 'landslide_category', 'landslide_trigger', 'landslide_size', 'landslide_setting', 'fatality_count', 'injury_count', 'country_name', 'country_code', 'admin_division_name', 'gazetteer_closest_point', 'gazetteer_distance', 'longitude', and 'latitude'.

The 'event_date' column, which initially contains both date and time information, is separated into two new columns: 'date' and 'time'. The separate_wider_delim() function from the tidyr package is used to split the 'event_date' column based on the space delimiter (' ').

The 'time' column is further separated into three new columns: 'hr', 'min', and 'sec', representing hours, minutes, and seconds, respectively. The separate_wider_delim() function is used again, this time splitting the 'time' column based on the colon delimiter (':').

The 'date' column is converted from a character format to a date format using the mdy() function from the lubridate package. The mdy() function assumes the date is in the format "month/day/year".

The 'hr', 'min', and 'sec' columns, which contain time components, are converted to numeric format using the as.numeric() function wrapped with mutate() and across() from the dplyr package.

A new 'date' column is created by combining the date and time components into a single datetime format. The make_datetime() function from the lubridate package is used, specifying the year, month, day, hour, minute, and second components extracted from the previously separated columns.

The ‘landslide_size’ column is converted to lowercase using the tolower() function wrapped with mutate().

The temporary ‘hr’, ‘min’, and ‘sec’ columns are removed from the dataset using the select() function with a negation (!) operator.

```
library(dplyr)
```

```
Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v forcats   1.0.0     v readr     2.1.5
v ggplot2   3.4.4     v stringr   1.5.1
v lubridate 1.9.3     v tibble    3.2.1
v purrr     1.0.2     v tidyr     1.3.0

-- 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 becor
```

```
library(readr)

data <- read_csv('data/nasa_global_landslide_catalog_point.csv',
                 show_col_types = FALSE) |>
  select(c('event_date','location_accuracy','landslide_category',
           'landslide_trigger', 'landslide_size',   'landslide_setting',
           'fatality_count', 'injury_count','country_name', 'country_code',
           'admin_division_name', 'gazetteer_closest_point', 'gazetteer_distance',
```

```
           'longitude', 'latitude')) |>
  separate_wider_delim(cols = 'event_date', ' ', names = c('date','time')) |>
  separate_wider_delim(cols = 'time', ':', names = c('hr','min','sec')) |>
  mutate(date = mdy(date)) |>
  mutate(across(c('hr','min','sec'),as.numeric)) |>
  mutate(date = make_datetime(year = year(date),
                              month = month(date),
                              day = day(date),
                              hour = hr,
                              min = min,
                              sec = sec)) |>
  mutate(landslide_size = tolower(landslide_size)) |>
  select(!c('hr','min','sec'))
write_csv(data, "cleaned_data.csv")
```

Other appendicies (as necessary)

```
deadly_size <- c("catastrophic", "very_large")
landsize_counts <- data |>
  group_by(admin_division_name) |>
  summarise(count_size = sum(landslide_size %in% deadly_size, na.rm = TRUE)) |>
  arrange(desc(count_size))
top_deadly_regions <- head(landsize_counts, 10)
print(top_deadly_regions)
```

```
# A tibble: 10 x 2
   admin_division_name count_size
   <chr>                    <int>
 1 Rio de Janeiro               8
 2 Aragua                       7
 3 California                   5
 4 Kerala                       5
 5 Xizang                       5
 6 Yunnan                       5
 7 Alaska                       4
 8 Sichuan                      4
 9 Gansu                        3
10 Uttaranchal                  3
```