/* Team Memeber : Payyavula Jaya Chandar and Oscar Lomibao Jr */

```r
library(rvest)
```

```
## Loading required package: xml2
```

```r
library('readr')
```

```
##
## Attaching package: 'readr'

## The following object is masked from 'package:rvest':
##
##     guess_encoding
```

```r
library('lubridate')
```

```
##
## Attaching package: 'lubridate'

## The following object is masked from 'package:base':
##
##     date
```

```r
library('plyr')
```

```
##
## Attaching package: 'plyr'

## The following object is masked from 'package:lubridate':
##
##     here
```

```r
library('dplyr')
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:plyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize

## The following objects are masked from 'package:lubridate':
##
##     intersect, setdiff, union

## The following objects are masked from 'package:stats':
##
##     filter, lag
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library('tidyr')
library(stringr)
# 1
solar_data <- read_html("https://www.spaceweatherlive.com/en/solar-activity/top-50-solar-flares")

# 2
solar_data <- solar_data %>%
 html_nodes("table") %>%
 .[[1]] %>%
 html_table() # 3
head('solar_data')
```

```
## [1] "solar_data"
```

```
# 4
colnames(solar_data)[1] <- "rank"
colnames(solar_data)[2] <- "x_class"
colnames(solar_data)[3] <- "date"
colnames(solar_data)[4] <- "region"
colnames(solar_data)[5] <- "start time"
colnames(solar_data)[6] <- "maximum time"
colnames(solar_data)[7] <- "end time"
colnames(solar_data)[8] <- "movie"
head('solar_data')
```

```
## [1] "solar_data"
```

```
# 5
tibble::as_tibble(solar_data)
```

```
## # A tibble: 50 × 8
##     rank x_class       date region `start time` `maximum time` `end time`
##    <int>   <chr>      <chr>  <chr>        <chr>          <chr>      <chr>
## 1      1   X28.0 2003/11/04   0486        19:29          19:53      20:06
## 2      2     X20 2001/04/02   9393        21:32          21:51      22:03
## 3      3   X17.2 2003/10/28   0486        09:51          11:10      11:24
## 4      4   X17.0 2005/09/07   0808        17:17          17:40      18:03
## 5      5   X14.4 2001/04/15   9415        13:19          13:50      13:55
## 6      6   X10.0 2003/10/29   0486        20:37          20:49      21:01
## 7      7    X9.4 1997/11/06      -        11:49          11:55      12:01
## 8      8    X9.0 2006/12/05   0930        10:18          10:35      10:45
## 9      9    X8.3 2003/11/02   0486        17:03          17:25      17:39
## 10    10    X7.1 2005/01/20   0720        06:36          07:01      07:26
## # ... with 40 more rows, and 1 more variables: movie <chr>
```

```
library(dplyr)
library(tidyr)
```

```
# 1
solar_data[8] <- NULL

# 2
solar_data_out1 <- tidyr::unite(solar_data, "start_datetime", 3, 5, sep = " ", remove = FALSE)
solar_data_out1 <- tidyr::unite(solar_data_out1, "max_datetime", 4, 7, sep = " ", remove = FALSE)
solar_data_out1 <- tidyr::unite(solar_data_out1, "end_datetime", 5, 9, sep = " ", remove = FALSE)
solar_data_out1[6] <- NULL
solar_data_out1[8] <- NULL
solar_data_out1[8] <- NULL
solar_data_out1[8] <- NULL
solar_data_out1[7] <- NULL

# 3
solar_data_out1 <- mutate(solar_data_out1,region = ifelse(stringr::str_detect(region, "-") , NA, region

# types
solar_data_out1 <- solar_data_out1 %>%
 readr::type_convert(col_types = cols(

 rank = col_integer(),
 start_datetime = col_datetime(format = "%Y/%m/%d %H:%M"),
 max_datetime = col_datetime(format = "%Y/%m/%d %H:%M"),
 end_datetime = col_datetime(format = "%Y/%m/%d %H:%M")
 ))
```

```
## Warning: The following named parsers don't match the column names: rank
```

```
tibble::as_tibble(solar_data_out1)
```

```
## # A tibble: 50 × 6
##     rank x_class      start_datetime        max_datetime
##    <int>   <chr>              <dttm>              <dttm>
## 1      1   X28.0 2003-11-04 19:29:00 2003-11-04 19:53:00
## 2      2     X20 2001-04-02 21:32:00 2001-04-02 21:51:00
## 3      3   X17.2 2003-10-28 09:51:00 2003-10-28 11:10:00
## 4      4   X17.0 2005-09-07 17:17:00 2005-09-07 17:40:00
## 5      5   X14.4 2001-04-15 13:19:00 2001-04-15 13:50:00
## 6      6   X10.0 2003-10-29 20:37:00 2003-10-29 20:49:00
## 7      7    X9.4 1997-11-06 11:49:00 1997-11-06 11:55:00
## 8      8    X9.0 2006-12-05 10:18:00 2006-12-05 10:35:00
## 9      9    X8.3 2003-11-02 17:03:00 2003-11-02 17:25:00
## 10    10    X7.1 2005-01-20 06:36:00 2005-01-20 07:01:00
## # ... with 40 more rows, and 2 more variables: end_datetime <dttm>,
## #   region <chr>
```

```
# 1
windwave_data <- read_html("http://cdaw.gsfc.nasa.gov/CME_list/radio/waves_type2.html")

# scrapes data from website
windwave_data <- windwave_data %>%
 html_nodes("pre") %>%
 .[1] %>%
 html_text()

# splits by new line
windwave_data <- stringr::str_split(windwave_data, '\n')

# converts list into data frame
df <- data.frame(as.list(windwave_data))

#deletes the unneccesary rows in data frame
df_new <- data.frame(df[13:494, ])

# 2
# creates a column name for the single column of data
colnames(df_new) <- c("one")

# divides the column into multiple columns, each having a name
df_out <- separate(df_new, one, c("start_date", "start_time", "end_date", "end_time", "start_frequency"

tibble::as_tibble(df_out)
```

```
## # A tibble: 482 × 14
##    start_date start_time end_date end_time start_frequency end_frequency
## *       <chr>      <chr>    <chr>    <chr>           <chr>         <chr>
## 1  1997/04/01      14:00    04/01    14:15            8000          4000
## 2  1997/04/07      14:30    04/07    17:30           11000          1000
## 3  1997/05/12      05:15    05/14    16:00           12000            80
## 4  1997/05/21      20:20    05/21    22:00            5000           500
## 5  1997/09/23      21:53    09/23    22:16            6000          2000
## 6  1997/11/03      05:15    11/03    12:00           14000           250
## 7  1997/11/03      10:30    11/03    11:30           14000          5000
## 8  1997/11/04      06:00    11/05    04:30           14000           100
## 9  1997/11/06      12:20    11/07    08:30           14000           100
## 10 1997/11/27      13:30    11/27    14:00           14000          7000
## # ... with 472 more rows, and 8 more variables: flare_location <chr>,
## #   flare_region <chr>, flare_classification <chr>, cme_date <chr>,
## #   cme_time <chr>, cme_angle <chr>, cme_width <chr>, cme_speed <chr>
```

```
# 1
windwave2 <- tibble::as_tibble(df_out)

windwave2 <- mutate(windwave2,start_frequency = ifelse(stringr::str_detect(start_frequency, "[?]+") , N
windwave2 <- mutate(windwave2,end_frequency = ifelse(stringr::str_detect(end_frequency, "[?]+") , NA, e

windwave2 <- mutate(windwave2,flare_region = ifelse(stringr::str_detect(flare_region, "-") , NA, flare_
windwave2 <- mutate(windwave2,flare_classification = ifelse(stringr::str_detect(flare_classification, "

windwave2 <- mutate(windwave2,cme_date = ifelse(stringr::str_detect(cme_date, "-") , NA, cme_date))
windwave2 <- mutate(windwave2,cme_time = ifelse(stringr::str_detect(cme_time, "-") , NA, cme_time))
windwave2 <- mutate(windwave2,cme_angle = ifelse(stringr::str_detect(cme_angle, "-") , NA, cme_angle))
windwave2 <- mutate(windwave2,cme_width = ifelse(stringr::str_detect(cme_width, "-") , NA, cme_width))
windwave2 <- mutate(windwave2,cme_speed = ifelse(stringr::str_detect(cme_speed, "-") , NA, cme_speed))
```

```
# 2
#Create a new column that indicates if a row corresponds to a halo flare or not,
windwave2 <- mutate(windwave2,cme_halo = ifelse(stringr::str_detect(cme_angle, "Halo") , TRUE, FALSE))

# and then replace Halo entries in the cme_angle column as NA.
windwave2 <- mutate(windwave2,cme_angle = ifelse(stringr::str_detect(cme_angle, "Halo") , NA, cme_angle

# 3
# Create a new column that indicates if width is given as a lower bound
windwave2 <- mutate(windwave2,cme_width_lb = ifelse(stringr::str_detect(cme_width, "[>]") , TRUE, FALSE

# remove any non-numeric part of the width column.
windwave2 <- mutate(windwave2,cme_width = ifelse(stringr::str_detect(cme_width, "[>]") , gsub(">","",wi

windwave2 <- mutate(windwave2,end_time = ifelse(stringr::str_detect(end_time, "24:00") , "00:00", end_t


# 4
# Combine date and time columns for start, end and cme so they can be encoded as datetime objects.
windwave2out <- tidyr::unite(windwave2, "start_datetime", 1, 2, sep = " ", remove = TRUE)
windwave2out <- tidyr::unite(windwave2out, "end_datetime", 2, 3, sep = " ", remove = TRUE)
windwave2out <- tidyr::unite(windwave2out, "cme_datetime", 8, 9, sep = " ", remove = TRUE)

# Extract years and append to datetimes
years <- format(as.Date(windwave2out$start_datetime, format= "%Y/%m/%d%R"), "%Y")
final_years <- as.data.frame(years)
windwave2out <- mutate(windwave2out, years)

windwave2out <- tidyr::unite(windwave2out, "end_datetime", 14, 2, sep = "/", remove = FALSE)
windwave2out <- tidyr::unite(windwave2out, "cme_datetime", 15, 9, sep = "/", remove = TRUE)


# 5

windwave2out <- windwave2out %>%
 readr::type_convert(col_types = cols(
 start_datetime = col_datetime(format = "%Y/%m/%d %R"),
end_datetime = col_datetime(format = "%Y/%m/%d %R"),
cme_datetime = col_datetime(format = "%Y/%m/%d %R"),
start_frequency = col_integer(),
end_frequency = col_integer(),
 cme_angle = col_integer(),
 cme_width = col_integer(),
 cme_speed = col_integer(),
 cme_halo = col_logical(),
 cme_width_lb = col_logical()
 ))
```

```
## Warning: The following named parsers don't match the column names:
## cme_halo, cme_width_lb

## Warning in type_convert_col(char_cols[[i]], specs$cols[[i]],
## which(is_character)[i], :
```

```
## Warning in type_convert_col(char_cols[[i]], specs$cols[[i]],
## which(is_character)[i], :

## Warning in type_convert_col(char_cols[[i]], specs$cols[[i]],
## which(is_character)[i], :

## Warning in type_convert_col(char_cols[[i]], specs$cols[[i]],
## which(is_character)[i], :

## Warning in type_convert_col(char_cols[[i]], specs$cols[[i]],
## which(is_character)[i], :

## Warning in type_convert_col(char_cols[[i]], specs$cols[[i]],
## which(is_character)[i], :

## Warning in type_convert_col(char_cols[[i]], specs$cols[[i]],
## which(is_character)[i], :

## Warning in type_convert_col(char_cols[[i]], specs$cols[[i]],
## which(is_character)[i], :

## Warning in type_convert_col(char_cols[[i]], specs$cols[[i]],
## which(is_character)[i], :

## Warning in type_convert_col(char_cols[[i]], specs$cols[[i]],
## which(is_character)[i], :

## Warning in type_convert_col(char_cols[[i]], specs$cols[[i]],
## which(is_character)[i], :

## Warning in type_convert_col(char_cols[[i]], specs$cols[[i]],
## which(is_character)[i], :

## Warning in type_convert_col(char_cols[[i]], specs$cols[[i]],
## which(is_character)[i], :

## Warning in type_convert_col(char_cols[[i]], specs$cols[[i]],
## which(is_character)[i], :

## Warning in type_convert_col(char_cols[[i]], specs$cols[[i]],
## which(is_character)[i], :

## Warning in type_convert_col(char_cols[[i]], specs$cols[[i]],
## which(is_character)[i], :

## Warning in type_convert_col(char_cols[[i]], specs$cols[[i]],
## which(is_character)[i], :

## Warning in type_convert_col(char_cols[[i]], specs$cols[[i]],
## which(is_character)[i], :

## Warning in type_convert_col(char_cols[[i]], specs$cols[[i]],
## which(is_character)[i], :
```

```
## Warning in type_convert_col(char_cols[[i]], specs$cols[[i]],
## which(is_character)[i], :
```

```
## Warning in type_convert_col(char_cols[[i]], specs$cols[[i]],
## which(is_character)[i], : [461, 10]: expected no trailing characters, but
## got 'h'
```

```
tibble::as_tibble(windwave2out)
```

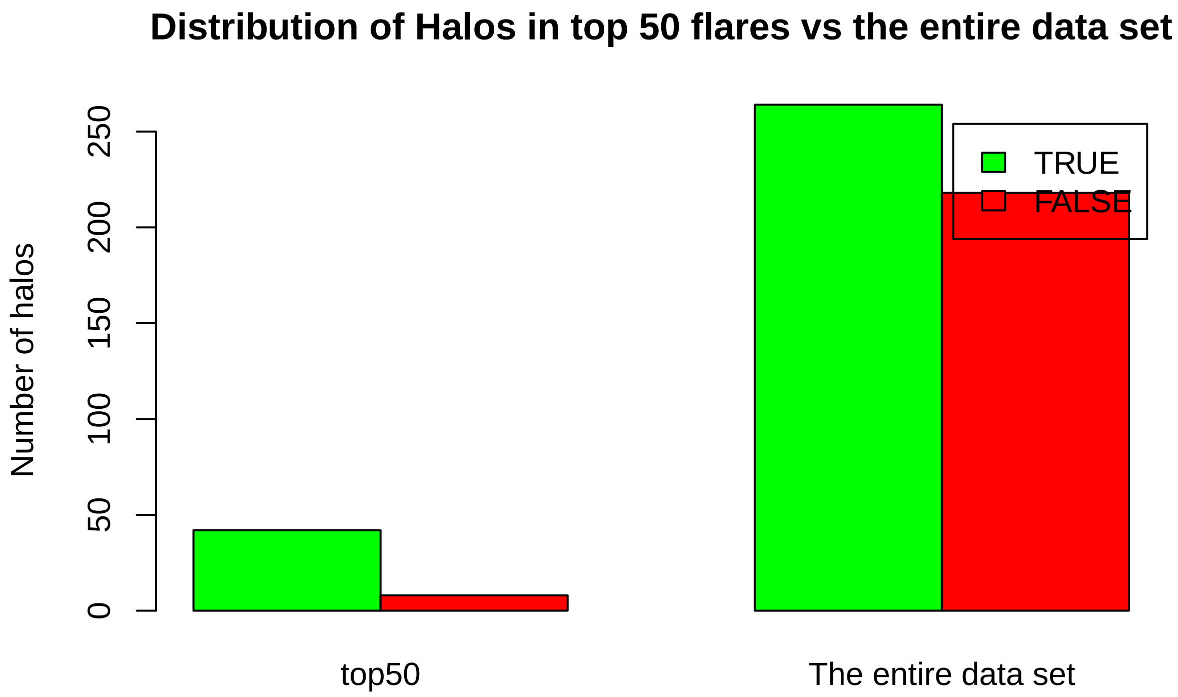
```
## # A tibble: 482 × 13
##         start_datetime        end_datetime start_frequency end_frequency
## *               <dttm>              <dttm>           <int>         <int>
## 1  1997-04-01 14:00:00 1997-04-01 14:15:00            8000          4000
## 2  1997-04-07 14:30:00 1997-04-07 17:30:00           11000          1000
## 3  1997-05-12 05:15:00 1997-05-14 16:00:00           12000            80
## 4  1997-05-21 20:20:00 1997-05-21 22:00:00            5000           500
## 5  1997-09-23 21:53:00 1997-09-23 22:16:00            6000          2000
## 6  1997-11-03 05:15:00 1997-11-03 12:00:00           14000           250
## 7  1997-11-03 10:30:00 1997-11-03 11:30:00           14000          5000
## 8  1997-11-04 06:00:00 1997-11-05 04:30:00           14000           100
## 9  1997-11-06 12:20:00 1997-11-07 08:30:00           14000           100
## 10 1997-11-27 13:30:00 1997-11-27 14:00:00           14000          7000
## # ... with 472 more rows, and 9 more variables: flare_location <chr>,
## #   flare_region <chr>, flare_classification <chr>, cme_angle <int>,
## #   cme_datetime <dttm>, cme_width <int>, cme_speed <int>, cme_halo <lgl>,
## #   cme_width_lb <lgl>
```

```
# 1
top_fifty <- windwave2out

# appends X numbers to the data frame w/o the X
top_fifty <- mutate(top_fifty,X = ifelse(stringr::str_detect(flare_classification, "[X]") , as.numeric(

# converts column into type double in order to rearrange
top_fifty <- top_fifty %>% type_convert(col_types = cols(X = col_double()))

#arranges in descending order
top_fifty <- arrange(top_fifty,desc(X))

final_top_fifty <- top_fifty
final_top_fifty <- final_top_fifty[-c(51:482), ]


# YES, we get data for the same solar flare events.
head(top_fifty)
```

```
## # A tibble: 6 × 14
##        start_datetime        end_datetime start_frequency end_frequency
##                <dttm>              <dttm>           <int>         <int>
## 1 2003-11-04 20:00:00 2003-11-04 00:00:00           10000           200
## 2 2001-04-02 22:05:00 2001-04-03 02:30:00           14000           250
## 3 2003-10-28 11:10:00 2003-10-29 00:00:00           14000            40
## 4 2001-04-15 14:05:00 2001-04-16 13:00:00           14000            40
## 5 2003-10-29 20:55:00 2003-10-29 00:00:00           11000           500
## 6 1997-11-06 12:20:00 1997-11-07 08:30:00           14000           100
## # ... with 10 more variables: flare_location <chr>, flare_region <chr>,
## #   flare_classification <chr>, cme_angle <int>, cme_datetime <dttm>,
## #   cme_width <int>, cme_speed <int>, cme_halo <lgl>, cme_width_lb <lgl>,
## #   X <dbl>
```

```
#2
# MODIFY SOLAR
new_solor_data <- solar_data
new_solor_data <- mutate(new_solor_data,region = ifelse(stringr::str_detect(region, "-") , NA, region))
colnames(new_solor_data)[3] <- "start_date"
colnames(new_solor_data)[4] <- "flare_region"
colnames(new_solor_data)[2] <- "flare_classification"

# MODIFY WINDWAVE
new_windwave2 <- windwave2
new_windwave2 <- mutate(new_windwave2,X = ifelse(stringr::str_detect(flare_classification, "[X]") , as.n
new_windwave2$start_date <- str_replace_all(new_windwave2$start_date, "-", "/")
new_windwave2 <- arrange(new_windwave2,desc(X))
new_windwave2 <- new_windwave2[-c(51:482), ]



comp_cols <- function(dframe1, dframe2) {
 sample <- merge(dframe1, dframe2)
 return(sample)
}
final_output <- comp_cols(new_windwave2, new_solor_data)

# Analysis Result : To analysize the best matching rows we intially modified both
# the datasets new_windwave2 and (top 50 of NASA) and new_solor_data(50 flares
# from SpaceWeatherLive.com) in such a way that both the dataframes have attributes of similar form.
# Then we wote a function that takes in 2 data frames and  merges them. The merge function
# is predefined function that combines rows of two different dataframes based on similar
# entities. The result of anaysis showed X2.6,X5.7,X4.0,X5.6 and X5.3 were same in both the data
# frames as in both the data frames the flare classification, region and start date were the same
# for these flares.Hence we got 5 matches. But to be precise, since we have a handful of matches
# we can observe that among them X2.6(region:8113) and X4.0(9236) had very close
# but not the same timmings and duration in both the dataframes.



#3
all_trues <- as.numeric(table(top_fifty$cme_halo)["TRUE"])
top_trues <- as.numeric(table(final_top_fifty$cme_halo)["TRUE"])
all_falses <- (length(top_fifty$cme_halo)) - all_trues
top_falses <- length((final_top_fifty$cme_halo)) - top_trues
one <- paste( toString(top_trues), toString(all_trues), sep=" ")
two <- paste( toString(top_falses), toString(all_falses), sep=" ")
final <- read.table(text = paste("A B", one, two, sep="\n"), header = TRUE)
```

```
names(final)[1] <- "top50"
names(final)[2] <- "The entire data set"
row.names(final)[1] <- "TRUE"
row.names(final)[2] <- "FALSE"
output <- barplot(as.matrix(final),main="Distribution of Halos in top 50 flares vs the entire data set"
ylab="Number of halos", col=c("green","red"),legend = rownames(final),beside = TRUE)
```


Distribution of Halos in top 50 flares vs the entire data set
Number of halos
🟩 TRUE
🟥 FALSE
top50
The entire data set