Project 1: Data scraping and cleaning

CMSC320, Spring 2019

Posted: Feb 19, 2019
Last Updated: Mar 12, 2019
Due: Mar 15, 2019

You’ve been hired by a new space weather startup looking to disrupt the space weather reporting business. Your first project is to provide better data about the top 50 solar flares recorded so far than the data shown by your startup ’s competitor SpaceWeatherLive.com. To do this, your boss pointed you to this messy HTML page from NASA (available here also) where you can get extra data that your startup can post in your new spiffy site.

Of course, you don’t have access to the raw data for either of these two tables, so as an enterprising data scientist you will scrape this information directly from each HTML page using all the great tools we have seen in class.

By the way, you should read up a bit on Solar Flares, coronal mass ejections, the solar flare alphabet soup, the scary storms of Halloween 2003, and sickening solar flares.

For this project you will do the following:

- Scrape and prepare each of the two datasets
- Integrate the two datasets (including some Entity Resolution)
- Exploratory Analysis

Part 1: Data scraping and preparation

Step 1: Scrape your competitor’s data (10 pts)

Scrape data for the top 50 solar flares shown in SpaceWeatherLive.com. Steps to do this are:

1. Use the `html` function to read the html page from the url above
2. Use the `html_nodes` function to find the page node corresponding to the table we want. Notice there are more than one table in that page, so you need to make sure you are finding the proper node.
3. Use the `html_table` to parse the table into a data frame
4. Rename attributes to some reasonable names for, e.g., `rank`, `flare_classification`, `date`, `flare_region`, `start_time`, `maximum_time`, `end_time`, `movie`. You can use `magrittr::set_colnames` within a pipeline.
5. Finish the pipeline with a call to `as_data_frame` to make the data frame easier to use.

The result should be a data frame, a `tibble` really, with the first few rows as:

```
# A tibble: 50 x 8
    rank flare_classification date       flare_region
   <int> <chr>                <chr>             <int>
 1     1 X28.0                2003/11/04          486
 2     2 X20.0                2001/04/02         9393
 3     3 X17.2                2003/10/28          486
 4     4 X17.0                2005/09/07          808
 5     5 X14.4                2001/04/15         9415
 6     6 X10.0                2003/10/29          486
 7     7 X9.4                 1997/11/06         8100
 8     8 X9.3                 2017/09/06         2673
 9     9 X9.0                 2006/12/05          930
10    10 X8.3                 2003/11/02          486
# ... with 40 more rows, and 4 more variables:
#   start_time <chr>, max_time <chr>, end_time <chr>,
#   movie <chr>
```

Step 2: Tidy the top 50 solar flare data (10 pts)

Your next step is to make sure this table is usable using the `dplyr` and `tidyr` packages:

1. Drop the last column of the table, since we are not going to use it moving forward.
2. Use the `tidyr::unite` package to combine the `date` and each of the three time columns into three datetime columns. You will see why this is useful later on.
3. Use the `readr::type_convert` function to convert columns containing datetimes into actual datetime objects (see `?col_datetime`, especially the `format` argument) .

The result of this step should be a data_frame with the first few rows as:

```
# A tibble: 50 x 6
    rank flare_classification start_datetime
 * <int> <chr>                <dttm>
 1      1 X28.0                2003-11-04 19:29:00
 2      2 X20.0                2001-04-02 21:32:00
 3      3 X17.2                2003-10-28 09:51:00
 4      4 X17.0                2005-09-07 17:17:00
 5      5 X14.4                2001-04-15 13:19:00
 6      6 X10.0                2003-10-29 20:37:00
 7      7 X9.4                 1997-11-06 11:49:00
 8      8 X9.3                 2017-09-06 11:53:00
 9      9 X9.0                 2006-12-05 10:18:00
10     10 X8.3                 2003-11-02 17:03:00
# ... with 40 more rows, and 3 more variables:
#   max_datetime <dttm>, end_datetime <dttm>,
#   flare_region <int>
```

Step 3. Scrape the NASA data (15 pts)

Next you need to scrape the data in http://cdaw.gsfc.nasa.gov/CME_list/radio/waves_type2.html (also available here) to get additional data about these solar flares. This table format is described here: https://cdaw.gsfc.nasa.gov/CME_list/radio/waves_type2_description.htm, and here:

NASA data description

The Wind/WAVES type II burst catalog: A brief description

URL: http://cdaw.gsfc.nasa.gov/CME_list/radio/waves_type2.html.

This is a catalog of type II bursts observed by the Radio and Plasma Wave (WAVES) experiment on board the Wind spacecraft and the associated coronal mass ejections (CMEs) observed by the Solar and Heliospheric Observatory (SOHO) mission. The type II burst catalog is derived from the Wind/WAVES catalog available at http://lep694.gsfc.nasa.gov/waves/waves.html by adding a few missing events.

The CMEs in this catalog are called radio-loud CMEs because of their ability to produce type II radio bursts. The CME sources are also listed, as derived from the Solar Geophysical Data listing or from inner coronal images such as Yohkoh/SXT and SOHO/EIT. Some solar sources have also been obtained from Solarsoft Latest Events Archive after October 1, 2002: http://www.lmsal.com/solarsoft/latest_events_archive.html

Explanation of catalog entries:

Column 1: Starting date of the type II burst (yyyy/mm/dd format)

Column 2: Starting time (UT) of the type II burst (hh:mm format)

Column 3: Ending date of the type II burst (mm/dd format; year in Column 1 applies)

Column 4: Ending time of the Type II burst (hh:mm format)

Column 5: Starting frequency of type II burst (kHz) [1]

Column 6: Ending frequency of type II burst (kHz) [1]

Column 7: Solar source location (Loc) of the associated eruption in heliographic coordinates [2]

Column 8: NOAA active region number (NOAA) [3]

Column 9: Soft X-ray flare importance (Imp) [4]

Column 10: Date of the associated CME (mm/dd format, Year in Column 1 applies) [5]

Column 11: Time of the associated CME (hh:mm format)

Column 12: Central position angle (CPA, degrees) for non-halo CMEs [6]

Column 13: CME width in the sky plane (degrees) [7]

Column 14: CME speed in the sky plane (km/s)

Column 15: Link to the daily proton, height-time, X-ray (PHTX) plots [8]

Notes

[1] ???? indicate that the starting and ending frequencies are not determined.

[2] Heliographic coordinates. S25E16 means the latitude is 25 deg south and 16 deg east (source located in the southeast quadrant of the Sun. N denotes northern latitudes and W denotes western longitudes. Entries like SW90 indicate that the source information is not complete, but we can say that the eruption occurs on the west limb but at southern latitudes; if such entries have a subscript b (e.g., NE90b) it means that the source is behind the particular limb. This information is usually gathered from SOHO/EIT difference images, which show dimming above the limb in question. Completely backside events with no information on the source location are marked as "back".

[3] If the active region number is not available or if the source region is not an active region, the entry is "—-". Filament regions are denoted by "FILA" or "DSF" for disappearing solar filament.

[4] Soft X-ray flare size (peak flux in the 1-8 A channel) from GOES. "—-" means the soft X-ray flux is not available.

[5] Lack of SOHO observations are noted as "LASCO DATA GAP". Other reasons are also noted if there is no CME parameters measured.

[6] The central position angle (CPA) is meaningful only for non-halo CMEs. For halo CMEs, the entry is "Halo". For halo CMEs, the height-time measurements are made at a position angle where the halo appears to move the fastest. This is known as the measurement position angle (MPA) and can be found in the main catalog (http://cdaw.gsfc.nasa.gov/CME_List).

[7] Width = 360 means the CME is a fill halo (see [6]). For some entries, there is a prefix ">", which means the reported width is a lower limit.

[8] 'PHTX' (proton, height-time, X-ray) link to three-day overview plots of solar energetic particle events (protons in the >10, >50 and >100 MeV GOES channels).

Links:

The CMEs and the type II bursts can be viewed together using the c2rdif_waves.html movies linked to the starting frequency (Column 5). The c3rdif_waves.html movies are linked to the ending frequencies (Column 6). The CMEs and the GOES flare light curves for a given type II burst can be viewed from the Javascript movies linked to the CME date (Column 10). The height-time plots (linear and quadratic) of the CMEs are linked to the CME speed (Column 14).

PHTX plots are linked to Column 15.

If you have questions, contact: Nat Gopalswamy (gopals@ssedmail.gsfc.nasa.gov)

This work is supported by NASA's Virtual Observatories Program

Tasks

1. Use `rvest` functions (e.g., `html_node`, `html_text`) and `stringr` functions (e.g., `str_split`, `str_subset`) to obtain each row of data as a long string. Create a `data_frame` using `as_data_frame` at this point so it's easier to use `dplyr` and `tidyr` for the next few steps.
2. Use `tidyr::separate` to separate each line of text into a data row. Choose appropriate names for columns.

The result of this step should be similar to:

```
# A tibble: 482 x 14
   start_date start_time end_date
 * <chr>      <chr>      <chr>
 1 1997/04/01 14:00      04/01
 2 1997/04/07 14:30      04/07
 3 1997/05/12 05:15      05/14
 4 1997/05/21 20:20      05/21
 5 1997/09/23 21:53      09/23
 6 1997/11/03 05:15      11/03
 7 1997/11/03 10:30      11/03
 8 1997/11/04 06:00      11/05
 9 1997/11/06 12:20      11/07
10 1997/11/27 13:30      11/27
# ... with 472 more rows, and 11 more
#   variables: end_time <chr>,
#   start_frequency <chr>, end_frequency
#   <chr>, flare_location <chr>,
#   flare_region <chr>,
#   flare_classification <chr>, cme_date
#   <chr>, cme_time <chr>, cme_angle
#   <chr>, cme_width <chr>, cme_speed
#   <chr>
```

Step 4: Tidy the NASA the table (15 pts)

Now, we tidy up the NASA table. Here we will code missing observations properly, recode columns that correspond to more than one piece of information, treat dates and times appropriately, and finally convert each column to the appropriate data type.

1. Recode any missing entries as `NA`. Refer to the data description in http://cdaw.gsfc.nasa.gov/CME_list/radio/waves_type2_description.html (and above) to see how missing entries are encoded.
2. The CPA column (`cme_angle`) contains angles in degrees for most rows, except for halo flares, which are coded as `Halo`. Create a new (logical) column that indicates if a row corresponds to a halo flare or not, and then replace `Halo` entries in the `cme_angle` column as NA.
3. The `width` column indicates if the given value is a lower bound. Create a new (logical) column that indicates if width is given as a lower bound, and remove any non-numeric part of the width column.
4. Combine date and time columns for `start`, `end` and `cme` so they can be encoded as `datetime` objects.
5. Use the `readr::type_convert` function to convert columns to appropriate types.

The output of this step should be similar to this (note the column types)

```
# A tibble: 482 x 13
   start_datetime      end_datetime
   <dttm>              <dttm>
 1 1997-04-01 14:00:00 1997-04-01 14:15:00
 2 1997-04-07 14:30:00 1997-04-07 17:30:00
 3 1997-05-12 05:15:00 1997-05-14 16:00:00
 4 1997-05-21 20:20:00 1997-05-21 22:00:00
 5 1997-09-23 21:53:00 1997-09-23 22:16:00
 6 1997-11-03 05:15:00 1997-11-03 12:00:00
 7 1997-11-03 10:30:00 1997-11-03 11:30:00
 8 1997-11-04 06:00:00 1997-11-05 04:30:00
 9 1997-11-06 12:20:00 1997-11-07 08:30:00
10 1997-11-27 13:30:00 1997-11-27 14:00:00
# ... with 472 more rows, and 11 more variables:
#   cme_datetime <dttm>, start_frequency <int>,
#   end_frequency <int>, flare_location <chr>,
#   flare_region <chr>, flare_classification <chr>,
#   cme_angle <int>, cme_speed <int>, halo <lgl>,
#   cme_width <int>, cme_width_limit <lgl>
```

Part 2: Analysis

Now that you have data from both sites, let's start some analysis.

Question 1: Replication (10 pts)

Can you replicate the top 50 solar flare table in SpaceWeatherLive.com exactly using the data obtained from NASA? That is, if you get the top 50 solar flares from the NASA table based on their classification (e.g., `X28` is the highest), do you get data for the same 50 solar flare events in the SpaceWeatherLive page? If not, why not?

Include code used to get the top 50 solar flares from the NASA table (be careful when ordering by classification, using `tidyr::separate` here is useful). Write a sentence or two discussing how well you can replicate the SpaceWeatherLive data from the NASA data.

Question 2: Entity Resolution (15 pts)

Let's see if we can improve how well the two datasets match each other by doing some *Entity Resolution*. Let's denote entities (flares) from the SpaceWeatherLive page as $E_1 E_1$, and entities (flares) from the NASA data as $E_2 E_2$

1. Write a function `flare_similarity` which computes a *similarity* $s(e_1, e_2) s(e_1, e_2)$ between flares $e_1 \in E_1 e_1 \in E_1$ and $e_2 \in E_2 e_2 \in E_2$.
2. Write a second function `flare_match` that computes for each flare $e_1 \in E_1 e_1 \in E_1$ which flare $e_2 \in E_2 e_2 \in E_2$ is the most similar. Your function can return `NA` if there is no entity $e_2 e_2$ that is sufficiently similar. Here, you determine what is the best matching entry in the NASA data for each of the top 50 solar flares in the SpaceWeatherLive page, if there is such an entry.
3. Add the result of `flare_match` to the top 50 table as the *index* of the best matching row in the NASA table, or `NA`.

In your submission, include an text explanation of how you define the similarity function $s(e_1, e_2) s(e_1, e_2)$ and how you use it to determine the best matching entitiy

Question 3: Analysis (10 pts)

Prepare one plot that shows the top 50 solar flares in context with all data available in the NASA dataset. Here are some possibilities (you can do something else)

1. Plot attributes in the NASA dataset (e.g., starting or ending frequencies, flare height or width) over time. Use graphical elements (e.g., text or points) to indicate flares in the top 50 classification.
2. Do flares in the top 50 tend to have Halo CMEs? You can make a barplot that compares the number (or proportion) of Halo CMEs in the top 50 flares vs. the dataset as a whole.
3. Do strong flares cluster in space? Are there solar regions that have strong flares (in the top 50) more commonly than expected (considering the full flare dataset)?

Submission

Prepare and knit an Rmarkdown file that includes for each step in Part 1: (a) code to carry out the step discussed, (b) partial output showing the output of your code, similar to the examples above, and (c) a short prose description of how your code works.

For questions 1 and 2 of Part 2, follow the instructions there.

For Question 3 of part 2 provide: (a) a short description (2 sentences) of what the intent of your plot is (think in terms of our discussion on how we show variation, co-variation in terms of central trend, spread, skew etc.), (b) code to produce your plot, (c) a short text description of your plot, and (d) a sentence or two of interpretation of your plot (again think of variation, co-variation, etc.).

Loading [MathJax]/jax/output/HTML-CSS/jax.js