CMSC320_P1

February 24, 2022

```
[1357]: #Part 1
```

```
[1358]: #Step 1
```

```
[1359]: import requests
        from bs4 import BeautifulSoup
        from urllib.parse import urlparse
        import pandas as pd
        import numpy as np
```

```
[1360]: res = requests.get("https://cmsc320.github.io/files/top-50-solar-flares.html")
        #get request
```

```
[1361]: root = BeautifulSoup(res.content)
```

```
[1362]: a = root.find("table")
```

```
[1363]: b = a.prettify()
```

```
[1364]: all_frames = pd.read_html(b)
        #use read_html to parser the html file
```

```
[1365]: all_frames[0].columns = ['rank', 'x_classification', 'date', 'region',␣
          ↪'start_time', 'maximum_time', 'end_time', 'movie']
        df = pd.DataFrame(all_frames[0])
        #Build a dataframe
```

```
[1366]: df
```

```
[1366]:     rank x_classification        date  region start_time maximum_time  \
        0      1             X28+  2003/11/04     486      19:29        19:53
        1      2             X20+  2001/04/02    9393      21:32        21:51
        2      3           X17.2+  2003/10/28     486      09:51        11:10
        3      4             X17+  2005/09/07     808      17:17        17:40
        4      5            X14.4  2001/04/15    9415      13:19        13:50
        5      6              X10  2003/10/29     486      20:37        20:49
        6      7             X9.4  1997/11/06    8100      11:49        11:55
```

```
7      8             X9.3  2017/09/06    2673      11:53        12:02
8      9               X9  2006/12/05     930      10:18        10:35
9     10             X8.3  2003/11/02     486      17:03        17:25
10    11             X8.2  2017/09/10    2673      15:35        16:06
11    12             X7.1  2005/01/20     720      06:36        07:01
12    13             X6.9  2011/08/09    1263      07:48        08:05
13    14             X6.5  2006/12/06     930      18:29        18:47
14    15             X6.2  2005/09/09     808      19:13        20:04
15    16             X6.2  2001/12/13    9733      14:20        14:30
16    17             X5.7  2000/07/14    9077      10:03        10:24
17    18             X5.6  2001/04/06    9415      19:10        19:21
18    19             X5.4  2012/03/07    1429      00:02        00:24
19    20             X5.4  2005/09/08     808      20:52        21:06
20    21             X5.4  2003/10/23     486      08:19        08:35
21    22             X5.3  2001/08/25    9591      16:23        16:45
22    23             X4.9  2014/02/25    1990      00:39        00:49
23    24             X4.9  1998/08/18    8307      22:10        22:19
24    25             X4.8  2002/07/23      39      00:18        00:35
25    26               X4  2000/11/26    9236      16:34        16:48
26    27             X3.9  2003/11/03     488      09:43        09:55
27    28             X3.9  1998/08/19    8307      21:35        21:45
28    29             X3.8  2005/01/17     720      06:59        09:52
29    30             X3.7  1998/11/22    8384      06:30        06:42
30    31             X3.6  2005/09/09     808      09:42        09:59
31    32             X3.6  2004/07/16     649      13:49        13:55
32    33             X3.6  2003/05/28     365      00:17        00:27
33    34             X3.4  2006/12/13     930      02:14        02:40
34    35             X3.4  2001/12/28    9767      20:02        20:45
35    36             X3.3  2013/11/05    1890      22:07        22:12
36    37             X3.3  2002/07/20      39      21:04        21:30
37    38             X3.3  1998/11/28    8395      04:54        05:52
38    39             X3.2  2013/05/14    1748      00:00        01:11
39    40             X3.1  2014/10/24    2192      21:07        21:41
40    41             X3.1  2002/08/24      69      00:49        01:12
41    42               X3  2002/07/15      30      19:59        20:08
42    43             X2.8  2013/05/13    1748      15:48        16:05
43    44             X2.8  2001/12/11    9733      07:58        08:08
44    45             X2.8  1998/08/18    8307      08:14        08:24
45    46             X2.7  2015/05/05    2339      22:05        22:11
46    47             X2.7  2003/11/03     488      01:09        01:30
47    48             X2.7  1998/05/06    8210      07:58        08:09
48    49             X2.6  2005/01/15     720      22:25        23:02
49    50             X2.6  2001/09/24    9632      09:32        10:38

   end_time                movie
0     20:06  Movie  View archive
1     22:03  Movie  View archive
```

2	11:24	Movie	View archive
3	18:03	Movie	View archive
4	13:55	Movie	View archive
5	21:01	Movie	View archive
6	12:01	Movie	View archive
7	12:10	Movie	View archive
8	10:45	Movie	View archive
9	17:39	Movie	View archive
10	16:31	Movie	View archive
11	07:26	Movie	View archive
12	08:08	Movie	View archive
13	19:00	Movie	View archive
14	20:36	Movie	View archive
15	14:35	Movie	View archive
16	10:43	Movie	View archive
17	19:31	Movie	View archive
18	00:40	Movie	View archive
19	21:17	Movie	View archive
20	08:49	Movie	View archive
21	17:04	Movie	View archive
22	01:03	Movie	View archive
23	22:28		View archive
24	00:47	Movie	View archive
25	16:56	Movie	View archive
26	10:19	Movie	View archive
27	21:50		View archive
28	10:07	Movie	View archive
29	06:49	Movie	View archive
30	10:08	Movie	View archive
31	14:01	Movie	View archive
32	00:39	Movie	View archive
33	02:57	Movie	View archive
34	21:32	Movie	View archive
35	22:15	Movie	View archive
36	21:54	Movie	View archive
37	06:13	Movie	View archive
38	01:20	Movie	View archive
39	22:13	Movie	View archive
40	01:31	Movie	View archive
41	20:14	Movie	View archive
42	16:16	Movie	View archive
43	08:14	Movie	View archive
44	08:32		View archive
45	22:15	Movie	View archive
46	01:45	Movie	View archive
47	08:20	Movie	View archive
48	23:31	Movie	View archive

```
49    11:09  Movie  View archive
```

[1367]:
```
#Step 2
```

[1368]:
```
df2 = df
df2 = df2.drop(['movie'],axis = 1)
#remove movie as mentioned in description
```

[1369]:
```
df2['start_datetime'] = pd.to_datetime(
                          df2['date'] + ' ' + df2['start_time'])
df2['max_datetime'] = pd.to_datetime(
                          df2['date'] + ' ' + df2['maximum_time'])
df2['end_datetime'] = pd.to_datetime(
                          df2['date'] + ' ' + df2['end_time'])
#Create three datetime columns by to_datetime
df2 = df2.drop(['date'],axis = 1)
df2 = df2.drop(['start_time'],axis = 1)
df2 = df2.drop(['maximum_time'],axis = 1)
df2 = df2.drop(['end_time'],axis = 1)
#Delete used columns
```

[1370]:
```
df2 = df2[['rank', 'x_classification', 'start_datetime', 'max_datetime',
      'end_datetime', 'region']]
#re-arrange the order of columns to better fit the sample output
```

[1371]:
```
df2.index = np.arange(1, 51)
# change index starting from 1
```

[1372]:
```
df2
```

[1372]:
```
    rank x_classification      start_datetime        max_datetime  \
1      1            X28+ 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+ 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 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            X9.3 2017-09-06 11:53:00 2017-09-06 12:02:00
9      9              X9 2006-12-05 10:18:00 2006-12-05 10:35:00
10    10            X8.3 2003-11-02 17:03:00 2003-11-02 17:25:00
11    11            X8.2 2017-09-10 15:35:00 2017-09-10 16:06:00
12    12            X7.1 2005-01-20 06:36:00 2005-01-20 07:01:00
13    13            X6.9 2011-08-09 07:48:00 2011-08-09 08:05:00
14    14            X6.5 2006-12-06 18:29:00 2006-12-06 18:47:00
15    15            X6.2 2005-09-09 19:13:00 2005-09-09 20:04:00
16    16            X6.2 2001-12-13 14:20:00 2001-12-13 14:30:00
```

```
17    17              X5.7 2000-07-14 10:03:00 2000-07-14 10:24:00
18    18              X5.6 2001-04-06 19:10:00 2001-04-06 19:21:00
19    19              X5.4 2012-03-07 00:02:00 2012-03-07 00:24:00
20    20              X5.4 2005-09-08 20:52:00 2005-09-08 21:06:00
21    21              X5.4 2003-10-23 08:19:00 2003-10-23 08:35:00
22    22              X5.3 2001-08-25 16:23:00 2001-08-25 16:45:00
23    23              X4.9 2014-02-25 00:39:00 2014-02-25 00:49:00
24    24              X4.9 1998-08-18 22:10:00 1998-08-18 22:19:00
25    25              X4.8 2002-07-23 00:18:00 2002-07-23 00:35:00
26    26                X4 2000-11-26 16:34:00 2000-11-26 16:48:00
27    27              X3.9 2003-11-03 09:43:00 2003-11-03 09:55:00
28    28              X3.9 1998-08-19 21:35:00 1998-08-19 21:45:00
29    29              X3.8 2005-01-17 06:59:00 2005-01-17 09:52:00
30    30              X3.7 1998-11-22 06:30:00 1998-11-22 06:42:00
31    31              X3.6 2005-09-09 09:42:00 2005-09-09 09:59:00
32    32              X3.6 2004-07-16 13:49:00 2004-07-16 13:55:00
33    33              X3.6 2003-05-28 00:17:00 2003-05-28 00:27:00
34    34              X3.4 2006-12-13 02:14:00 2006-12-13 02:40:00
35    35              X3.4 2001-12-28 20:02:00 2001-12-28 20:45:00
36    36              X3.3 2013-11-05 22:07:00 2013-11-05 22:12:00
37    37              X3.3 2002-07-20 21:04:00 2002-07-20 21:30:00
38    38              X3.3 1998-11-28 04:54:00 1998-11-28 05:52:00
39    39              X3.2 2013-05-14 00:00:00 2013-05-14 01:11:00
40    40              X3.1 2014-10-24 21:07:00 2014-10-24 21:41:00
41    41              X3.1 2002-08-24 00:49:00 2002-08-24 01:12:00
42    42                X3 2002-07-15 19:59:00 2002-07-15 20:08:00
43    43              X2.8 2013-05-13 15:48:00 2013-05-13 16:05:00
44    44              X2.8 2001-12-11 07:58:00 2001-12-11 08:08:00
45    45              X2.8 1998-08-18 08:14:00 1998-08-18 08:24:00
46    46              X2.7 2015-05-05 22:05:00 2015-05-05 22:11:00
47    47              X2.7 2003-11-03 01:09:00 2003-11-03 01:30:00
48    48              X2.7 1998-05-06 07:58:00 1998-05-06 08:09:00
49    49              X2.6 2005-01-15 22:25:00 2005-01-15 23:02:00
50    50              X2.6 2001-09-24 09:32:00 2001-09-24 10:38:00

          end_datetime  region
1  2003-11-04 20:06:00     486
2  2001-04-02 22:03:00    9393
3  2003-10-28 11:24:00     486
4  2005-09-07 18:03:00     808
5  2001-04-15 13:55:00    9415
6  2003-10-29 21:01:00     486
7  1997-11-06 12:01:00    8100
8  2017-09-06 12:10:00    2673
9  2006-12-05 10:45:00     930
10 2003-11-02 17:39:00     486
11 2017-09-10 16:31:00    2673
```

```
12 2005-01-20 07:26:00     720
13 2011-08-09 08:08:00    1263
14 2006-12-06 19:00:00     930
15 2005-09-09 20:36:00     808
16 2001-12-13 14:35:00    9733
17 2000-07-14 10:43:00    9077
18 2001-04-06 19:31:00    9415
19 2012-03-07 00:40:00    1429
20 2005-09-08 21:17:00     808
21 2003-10-23 08:49:00     486
22 2001-08-25 17:04:00    9591
23 2014-02-25 01:03:00    1990
24 1998-08-18 22:28:00    8307
25 2002-07-23 00:47:00      39
26 2000-11-26 16:56:00    9236
27 2003-11-03 10:19:00     488
28 1998-08-19 21:50:00    8307
29 2005-01-17 10:07:00     720
30 1998-11-22 06:49:00    8384
31 2005-09-09 10:08:00     808
32 2004-07-16 14:01:00     649
33 2003-05-28 00:39:00     365
34 2006-12-13 02:57:00     930
35 2001-12-28 21:32:00    9767
36 2013-11-05 22:15:00    1890
37 2002-07-20 21:54:00      39
38 1998-11-28 06:13:00    8395
39 2013-05-14 01:20:00    1748
40 2014-10-24 22:13:00    2192
41 2002-08-24 01:31:00      69
42 2002-07-15 20:14:00      30
43 2013-05-13 16:16:00    1748
44 2001-12-11 08:14:00    9733
45 1998-08-18 08:32:00    8307
46 2015-05-05 22:15:00    2339
47 2003-11-03 01:45:00     488
48 1998-05-06 08:20:00    8210
49 2005-01-15 23:31:00     720
50 2001-09-24 11:09:00    9632
```

```
[1373]: #Step 3
```

```
[1374]: res2 = requests.get("https://cmsc320.github.io/files/waves_type2.html")
```

```
[1375]: root = BeautifulSoup(res2.content)
```

[1376]:

```
root = root.text
root = root.split("\n")
```

[1377]:

```
arr_2d = []
for x in range(15,533):
    arr_2d.append(root[x].split(" "))
#separate each element
```

[1378]:

```
arr_2d = [[i for i in item if i != ''] for item in arr_2d]
#remove empty in arr_2d
```

[1379]:

```
df3 = pd.DataFrame(arr_2d)
#build dataframe
```

[1380]:

```
for i in range (15,24):
    df3 = df3.drop(i,axis = 1)
#drop unused columns
```

[1381]:

```
df3.columns = ['start_date <chr>', 'start_time <chr>', 'end_date <chr>',␣
 ↪'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>','plot <chr>']
#rename cloumns
```

[1382]:

```
df3
```

[1382]:

```
    start_date <chr> start_time <chr> end_date <chr> end_time <chr>  \
0         1997/04/01            14:00          04/01          14:15
1         1997/04/07            14:30          04/07          17:30
2         1997/05/12            05:15          05/14          16:00
3         1997/05/21            20:20          05/21          22:00
4         1997/09/23            21:53          09/23          22:16
..               …                …             …             …
513       2017/09/04            20:27          09/05          04:54
514       2017/09/06            12:05          09/07          08:00
515       2017/09/10            16:02          09/11          06:50
516       2017/09/12            07:38          09/12          07:43
517       2017/09/17            11:45          09/17          12:35

    start_frequency <chr> end_frequency <chr> flare_location <chr>  \
0                    8000                4000                S25E16
1                   11000                1000                S28E19
2                   12000                  80                N21W08
3                    5000                 500                N05W12
4                    6000                2000                S29E25
..                     …                   …                   …
```

```
513                 14000                 210               S10W12
514                 16000                  70               S08W33
515                 16000                 150               S09W92
516                 16000               13000               N08E48
517                 16000                 900              S08E170

    flare_region <chr> flare_classification <chr> cme_date <chr>  \
0                 8026                       M1.3          04/01
1                 8027                       C6.8          04/07
2                 8038                       C1.3          05/12
3                 8040                       M1.3          05/21
4                 8088                       C1.4          09/23
..                 …                        …            …
513              12673                       M5.5          09/04
514              12673                       X9.3          09/06
515              -----                       X8.3          09/10
516              12680                       C3.0          09/12
517              -----                       ----          09/17

    cme_time <chr> cme_angle <chr> cme_width <chr> cme_speed <chr> plot <chr>
0            15:18              74              79             312       PHTX
1            14:27            Halo             360             878       PHTX
2            05:30            Halo             360             464       PHTX
3            21:00             263             165             296       PHTX
4            22:02             133             155             712       PHTX
..             …             …             …             …        …
513          20:12            Halo             360            1418       PHTX
514          12:24            Halo             360            1571       PHTX
515          16:00            Halo             360            3163       PHTX
516          08:03             124              96             252       PHTX
517          12:00            Halo             360            1385       PHTX

[518 rows x 15 columns]
```

```
[1383]: #Step 4
```

```
[1384]: df4 = df3
```

```
[1385]: for i in range(0,518):
             for j in range(0,14):
                 if df4.iat[i,j][0] == "-":
                     df4.iat[i,j] = "NaN"
         #Using for lopp to replace not avaible values by NaN
```

```
[1386]: for i in range(0,518):
             df4.iat[i,2] = df4.iat[i,0][0]+df4.iat[i,0][1]+df4.iat[i,0][2] + df4.
          ↪iat[i,0][3]+df4.iat[0,0][4]+ df4.iat[i,2]
```

```
    if df4.iat[i,9] != "NaN":
        df4.iat[i,9] = df4.iat[i,0][0]+df4.iat[i,0][1]+df4.iat[i,0][2] + df4.
 ↪iat[i,0][3]+df4.iat[0,0][4]+ df4.iat[i,9]
# Add year to end_date and cme_date then date has the form YYYY/MM/DD
# If cme_date is NaN, we skip
```

[1387]:

```
for i in range(0,518):
    df4.iat[i,1] = df4.iat[i,1]+":00"
    df4.iat[i,3] = df4.iat[i,3]+":00"
    if df4.iat[i,9] != "NaN":
        df4.iat[i,10] = df4.iat[i,10]+":00"
# Add second to start_time, end_time and cme_time then date has the form HH/MM/
 ↪SS
# If cme_date is NaN, we skip
```

[1388]:

```
df4['start_datetime'] = pd.to_datetime(df4['start_date <chr>']) + pd.
 ↪to_timedelta(df4['start_time <chr>'])
df4['end_datetime'] = pd.to_datetime(df4['end_date <chr>']) + pd.
 ↪to_timedelta(df4['end_time <chr>'])
df4['cme_datetime'] = pd.to_datetime(df4['cme_date <chr>']) + pd.
 ↪to_timedelta(df4['cme_time <chr>'])
# Create three datetime columns using to_datetime
```

[1389]:

```
df4.columns = ['start_date', 'start_time', 'end_date', 'end_time',␣
 ↪'start_frequency', 'end_frequency', 'flare_location', 'flare_region',
              'flare_classification', 'cme_date', 'cme_time', 'cme_angle',␣
 ↪'cme_width',␣
 ↪'cme_speed','plot','start_datetime','end_datetime','cme_datetime']
#rename columns
```

[1390]:

```
#lambda expression !!!!!
list1 = []
for i in range(0,518):
    if df4.iat[i,11] == "Halo":
        df4.iat[i,11] = "NA"
        list1.append(True)
    else :
        list1.append(False)
df4["is_halo"] = list1
# Create "is_halo" column by iterating all elements.
# Also repace all "Halo" to "NA"
```

[1391]:

```
list2 = []
for i in range(0,518):
    if df4.iat[i,12][0] == ">":
        df4.iat[i,12] = df4.iat[i,12][1:]
        list2.append(True)
    else :
        list2.append(False)
df4["width_lower_bound"] = list2
# Create "width_lower_bound" column by iterating all elements and checking if␣
 ↪there is a ">".
# Also delete ">" if there is one
```

[1392]:

```
df4 = df4.drop(['start_time'],axis = 1)
df4 = df4.drop(['start_date'],axis = 1)
df4 = df4.drop(['end_time'],axis = 1)
df4 = df4.drop(['end_date'],axis = 1)
df4 = df4.drop(['cme_time'],axis = 1)
df4 = df4.drop(['cme_date'],axis = 1)
# Drop columns to better fit with sample output
```

[1393]:

```
df4 = df4[['start_datetime','end_datetime','start_frequency', 'end_frequency',␣
 ↪'flare_location', 'flare_region',
              'flare_classification', 'cme_datetime', 'cme_angle', 'cme_width',␣
 ↪'cme_speed','plot','is_halo','width_lower_bound']]
# Change columns olders to better fit with sample output
```

[1394]:

```
df4
```

[1394]:

```
          start_datetime        end_datetime start_frequency end_frequency  \
0   1997-04-01 14:00:00 1997-04-01 14:15:00            8000          4000
1   1997-04-07 14:30:00 1997-04-07 17:30:00           11000          1000
2   1997-05-12 05:15:00 1997-05-14 16:00:00           12000            80
3   1997-05-21 20:20:00 1997-05-21 22:00:00            5000           500
4   1997-09-23 21:53:00 1997-09-23 22:16:00            6000          2000
..                  …                   …              …             …
513 2017-09-04 20:27:00 2017-09-05 04:54:00           14000           210
514 2017-09-06 12:05:00 2017-09-07 08:00:00           16000            70
515 2017-09-10 16:02:00 2017-09-11 06:50:00           16000           150
516 2017-09-12 07:38:00 2017-09-12 07:43:00           16000         13000
517 2017-09-17 11:45:00 2017-09-17 12:35:00           16000           900

    flare_location flare_region flare_classification        cme_datetime  \
0           S25E16         8026                 M1.3 1997-04-01 15:18:00
1           S28E19         8027                 C6.8 1997-04-07 14:27:00
2           N21W08         8038                 C1.3 1997-05-12 05:30:00
3           N05W12         8040                 M1.3 1997-05-21 21:00:00
4           S29E25         8088                 C1.4 1997-09-23 22:02:00
..             …            …                  …                   …
513         S10W12        12673                 M5.5 2017-09-04 20:12:00
514         S08W33        12673                 X9.3 2017-09-06 12:24:00
515         S09W92          NaN                 X8.3 2017-09-10 16:00:00
```

```
516         N08E48        12680                 C3.0 2017-09-12 08:03:00
517        S08E170          NaN                  NaN 2017-09-17 12:00:00

    cme_angle cme_width cme_speed  plot  is_halo  width_lower_bound
0          74        79       312  PHTX    False              False
1          NA       360       878  PHTX     True              False
2          NA       360       464  PHTX     True              False
3         263       165       296  PHTX    False              False
4         133       155       712  PHTX    False              False
..        …       …       …   …      …                …
513        NA       360      1418  PHTX     True              False
514        NA       360      1571  PHTX     True              False
515        NA       360      3163  PHTX     True              False
516       124        96       252  PHTX    False              False
517        NA       360      1385  PHTX     True              False

[518 rows x 14 columns]
```

[1395]:

```
#Part 2 Analysis
```

[1396]:

```
#Q1
```

[1397]:

```
list3 = []
#for this sorting, we convert flare_classification to weighted numbers.
#As researched, Classification has a letter which is A B C M X(increasing␣
 ↪order) and a number.
for i in range(0,518):
    if df4.iat[i,6] != "NaN" and df4.iat[i,6] != "FILA" and df4.iat[i,6] !=␣
 ↪"DSF":
        folat_str = df4.iat[i,6][1:]
        list3.append(1000*ord(df4.iat[i,6][0])+float(folat_str))
        #Thus, we can give the letter a weight of 1000 and the number weight of␣
 ↪1.
        #(Since the number is much less likely to influence the rank compared␣
 ↪to the letter.)
    else:
        list3.append(0)
        #If there is no data, we assign 0.
df5 = df4
df5["weighted_flare_rank"] = list3
#We added our new column to the table.
```

[1398]:

```
df5 = df5.drop(['start_frequency'], axis = 1)
df5 = df5.drop(['end_frequency'], axis = 1)
df5 = df5.drop(['flare_location'], axis = 1)
df5 = df5.drop(['cme_angle'], axis = 1)
df5 = df5.drop(['cme_width'], axis = 1)
```

```
df5 = df5.drop(['cme_speed'], axis = 1)
df5 = df5.drop(['plot'], axis = 1)
df5 = df5.drop(['is_halo'], axis = 1)
df5 = df5.drop(['width_lower_bound'], axis = 1)
df5 = df5.drop(['cme_datetime'], axis = 1)
#we delele unshown columns compared to SWL
```

[1399]:

```
df5.sort_values(by = 'weighted_flare_rank',ascending=False).head(50)
#We sort by the weighted_flare_rank in decending order and only pick top 50.
```

[1399]:

```
         start_datetime        end_datetime flare_region flare_classification  \
240 2003-11-04 20:00:00 2003-11-05 00:00:00        10486                 X28.
117 2001-04-02 22:05:00 2001-04-03 02:30:00         9393                 X20.
233 2003-10-28 11:10:00 2003-10-30 00:00:00        10486                 X17.
126 2001-04-15 14:05:00 2001-04-16 13:00:00         9415                 X14.
234 2003-10-29 20:55:00 2003-10-30 00:00:00        10486                 X10.
8   1997-11-06 12:20:00 1997-11-07 08:30:00         8100                 X9.4
514 2017-09-06 12:05:00 2017-09-07 08:00:00        12673                 X9.3
328 2006-12-05 10:50:00 2006-12-05 20:00:00        10930                 X9.0
515 2017-09-10 16:02:00 2017-09-11 06:50:00          NaN                 X8.3
237 2003-11-02 17:30:00 2003-11-03 01:00:00        10486                 X8.3
288 2005-01-20 07:15:00 2005-01-20 16:30:00        10720                 X7.1
359 2011-08-09 08:20:00 2011-08-09 08:35:00        11263                 X6.9
331 2006-12-06 19:00:00 2006-12-09 00:00:00        10930                 X6.5
317 2005-09-09 19:45:00 2005-09-09 22:00:00        10808                 X6.2
82  2000-07-14 10:30:00 2000-07-15 14:30:00         9077                 X5.7
121 2001-04-06 19:35:00 2001-04-07 01:50:00         9415                 X5.6
375 2012-03-07 01:00:00 2012-03-08 19:00:00        11429                 X5.4
135 2001-08-25 16:50:00 2001-08-25 23:00:00         9591                 X5.3
443 2014-02-25 00:56:00 2014-02-25 11:28:00        11990                 X4.9
193 2002-07-23 00:50:00 2002-07-23 04:00:00        10039                 X4.8
104 2000-11-26 17:00:00 2000-11-26 17:15:00         9236                 X4.0
239 2003-11-03 10:00:00 2003-11-03 12:30:00        10488                 X3.9
286 2005-01-17 10:00:00 2005-01-17 10:35:00        10720                 X3.8
222 2003-05-28 01:00:00 2003-05-29 00:30:00        10365                 X3.6
332 2006-12-13 02:45:00 2006-12-13 10:40:00        10930                 X3.4
160 2001-12-28 20:35:00 2001-12-29 03:00:00         9756                 X3.4
192 2002-07-20 21:30:00 2002-07-20 22:20:00        10039                 X3.3
404 2013-05-14 01:16:00 2013-05-14 08:20:00        11748                 X3.2
201 2002-08-24 01:45:00 2002-08-24 03:25:00        10069                 X3.1
403 2013-05-13 16:15:00 2013-05-13 19:10:00        11748                 X2.8
487 2015-05-05 22:24:00 2015-05-05 23:14:00        12339                 X2.7
19  1998-05-06 08:25:00 1998-05-06 08:35:00         8210                 X2.7
238 2003-11-03 01:15:00 2003-11-03 01:25:00        10488                 X2.7
284 2005-01-15 23:00:00 2005-01-17 00:00:00        10720                 X2.6
142 2001-09-24 10:45:00 2001-09-25 20:00:00         9632                 X2.6
9   1997-11-27 13:30:00 1997-11-27 14:00:00         8113                 X2.6
```

```
276 2004-11-10 02:25:00 2004-11-10 03:40:00        10696                 X2.5
73  2000-06-06 15:20:00 2000-06-08 09:00:00         9026                 X2.3
123 2001-04-10 05:24:00 2001-04-11 00:00:00         9415                 X2.3
99  2000-11-24 15:25:00 2000-11-24 22:00:00         9236                 X2.3
345 2011-02-15 02:10:00 2011-02-15 07:00:00        11158                 X2.2
318 2005-09-10 21:45:00 2005-09-11 01:00:00        10808                 X2.1
361 2011-09-06 22:30:00 2011-09-07 15:40:00        11283                 X2.1
420 2013-10-25 15:08:00 2013-10-25 22:32:00        11882                 X2.1
7   1997-11-04 06:00:00 1997-11-05 04:30:00         8100                 X2.1
98  2000-11-24 05:10:00 2000-11-24 15:00:00         9236                 X2.0
125 2001-04-12 10:20:00 2001-04-12 10:40:00         9415                 X2.0
274 2004-11-07 16:25:00 2004-11-08 20:00:00        10696                 X2.0
285 2005-01-17 09:25:00 2005-01-17 16:00:00        10720                 X2.0
102 2000-11-25 19:00:00 2000-11-25 19:35:00         9236                 X1.9

     weighted_flare_rank
240              88028.0
117              88020.0
233              88017.0
126              88014.0
234              88010.0
8                88009.4
514              88009.3
328              88009.0
515              88008.3
237              88008.3
288              88007.1
359              88006.9
331              88006.5
317              88006.2
82               88005.7
121              88005.6
375              88005.4
135              88005.3
443              88004.9
193              88004.8
104              88004.0
239              88003.9
286              88003.8
222              88003.6
332              88003.4
160              88003.4
192              88003.3
404              88003.2
201              88003.1
403              88002.8
487              88002.7
```

```
19                  88002.7
238                 88002.7
284                 88002.6
142                 88002.6
9                   88002.6
276                 88002.5
73                  88002.3
123                 88002.3
99                  88002.3
345                 88002.2
318                 88002.1
361                 88002.1
420                 88002.1
7                   88002.1
98                  88002.0
125                 88002.0
274                 88002.0
285                 88002.0
102                 88001.9
```

[1400]:
```
#We cannot replicate the data exactly. There are some missing solar flare␣
 ↪events.
#The data we get from NASA is not identical to SpaceWeatherLive but most rows␣
 ↪can
#replicate the SpaceWeatherLive by indetical flare_region and␣
 ↪flare_classification(times may differ).
#To be noticed, some regions have a leading "1" and some flare_classifications␣
 ↪have "+" at the end.
#We consider it is same when other parts are identical.
```

[1401]:
```
#Q2
```

[1402]:
```
df4_NASA = df5
df5_SWL = df2
```

[1403]:
```
#To find best matches, firstly compare the x-classification, region.
#If those are matched, we consider it is a match.
#If many match to the same row, we then compare the difference of their date␣
 ↪and time. We picked the smallest one.
#
```

[1404]:
```
rank = [-1]*518
time_difference = [-1]*518
#rank is the ranks in SWL for NASA values

#time_difference is the sum of absolute value of
```

```
#(SWl_start_datetime - NASA_start_datetime) and
# absolute value of (SWl_end_datetime - NASA_end_datetime)

#We firstly initialize ranks and time_difference to be -1, which means not avaible.

for i in range (0,50):
    if df5_SWL.iat[i,1][-1] == "+":
        SWL_x_class = df5_SWL.iat[i,1][:-1]
    else:
        SWL_x_class = df5_SWL.iat[i,1]
    #Since some X_class has a "+", we will remove that for comparision.

    SWL_region = df5_SWL.iat[i,5]
    SWl_start_datetime = df5_SWL.iat[i,2]
    SWl_end_datetime = df5_SWL.iat[i,4]
   #We get region,start and end for SWL data.

    for j in range(0,518):#(0,518)
        if df4_NASA.iat[j,3][-1] == ".":
            NASA_x_class = df4_NASA.iat[j,3][:-1]
        else:
            NASA_x_class = df4_NASA.iat[j,3]
            #Since some X_class has a ".", we will remove that for comparision.

        NASA_region = df4_NASA.iat[j,2]
        if(len(str(NASA_region)) == 5):
            NASA_region = int(str(NASA_region)[1:])
        #Since some region has a "1" at beginning, we will remove that for comparision.

        NASA_start_datetime = df4_NASA.iat[j,0]
        NASA_end_datetime = df4_NASA.iat[j,1]
        time_difference_new = abs(SWl_start_datetime - NASA_start_datetime) + \
        abs(SWl_end_datetime - NASA_end_datetime)

        if SWL_x_class == NASA_x_class and SWL_region == NASA_region :
        #If x_class and region matches, we think it is a match.
            if rank[j] != -1:
                #If the value already has a match, we compare by their time_difference.
                if time_difference_new < time_difference[j]:
                    #If new value has smaller time_difference, we update our rank and time_difference
                    rank[j] = i+1
                    time_difference[j] = time_difference_new
            else :
                #If the value doesn't have a match, we update our rank and␣
 ↪time_difference
                rank[j] = i+1
                time_difference[j] = time_difference_new
df4["rank"] = rank
#We add rank to the dataframe.
#df4["time_dif"] = time_difference
```

[1405]:

```
df4 = df4.replace(-1,"")
df4
```

[1405]:

```
         start_datetime        end_datetime start_frequency end_frequency  \
0   1997-04-01 14:00:00 1997-04-01 14:15:00            8000          4000
1   1997-04-07 14:30:00 1997-04-07 17:30:00           11000          1000
2   1997-05-12 05:15:00 1997-05-14 16:00:00           12000            80
3   1997-05-21 20:20:00 1997-05-21 22:00:00            5000           500
4   1997-09-23 21:53:00 1997-09-23 22:16:00            6000          2000
..                  …                 …             …           …
513 2017-09-04 20:27:00 2017-09-05 04:54:00           14000           210
514 2017-09-06 12:05:00 2017-09-07 08:00:00           16000            70
515 2017-09-10 16:02:00 2017-09-11 06:50:00           16000           150
516 2017-09-12 07:38:00 2017-09-12 07:43:00           16000         13000
517 2017-09-17 11:45:00 2017-09-17 12:35:00           16000           900

    flare_location flare_region flare_classification        cme_datetime  \
0           S25E16         8026                 M1.3 1997-04-01 15:18:00
1           S28E19         8027                 C6.8 1997-04-07 14:27:00
2           N21W08         8038                 C1.3 1997-05-12 05:30:00
3           N05W12         8040                 M1.3 1997-05-21 21:00:00
4           S29E25         8088                 C1.4 1997-09-23 22:02:00
..             …          …                  …                 …
513         S10W12        12673                 M5.5 2017-09-04 20:12:00
514         S08W33        12673                 X9.3 2017-09-06 12:24:00
515         S09W92          NaN                 X8.3 2017-09-10 16:00:00
516         N08E48        12680                 C3.0 2017-09-12 08:03:00
517        S08E170          NaN                  NaN 2017-09-17 12:00:00

    cme_angle cme_width cme_speed  plot  is_halo  width_lower_bound  \
0          74        79       312  PHTX    False              False
1          NA       360       878  PHTX     True              False
2          NA       360       464  PHTX     True              False
3         263       165       296  PHTX    False              False
4         133       155       712  PHTX    False              False
..        …       …       …   …      …                …
513        NA       360      1418  PHTX     True              False
514        NA       360      1571  PHTX     True              False
515        NA       360      3163  PHTX     True              False
```

```
516       124        96       252  PHTX     False                  False
517        NA       360      1385  PHTX      True                  False

     weighted_flare_rank rank
0                77001.3
1                67006.8
2                67001.3
3                77001.3
4                67001.4
..                   …   …
513              77005.5
514              88009.3    8
515              88008.3
516              67003.0
517                  0.0

[518 rows x 16 columns]
```

[1406]:

```python
# there is no duplicates SWL entries "best matches".
```

[1407]:

```python
#Q3
#Intention: To check if there is a relationship between higher X_classification and Halo CME.
#The variation is the x_classification, which means some of the strongest solar flares with others.
# We compare the height difference (or proportion) of Halo CMEs in the top 50 flares vs. the dataset as a whole.
```

[1408]:

```python
import matplotlib.pyplot as plt
```

[1409]:

```python
count_top50 = 0
count_top50_halo = 0
for i in range(0,518):
    if df4.iat[i,15]!= "" :
        count_top50 += 1
        if df4.iat[i,12] == True:
            count_top50_halo +=1
print(count_top50)
print(count_top50_halo)
```

```
22
19
```

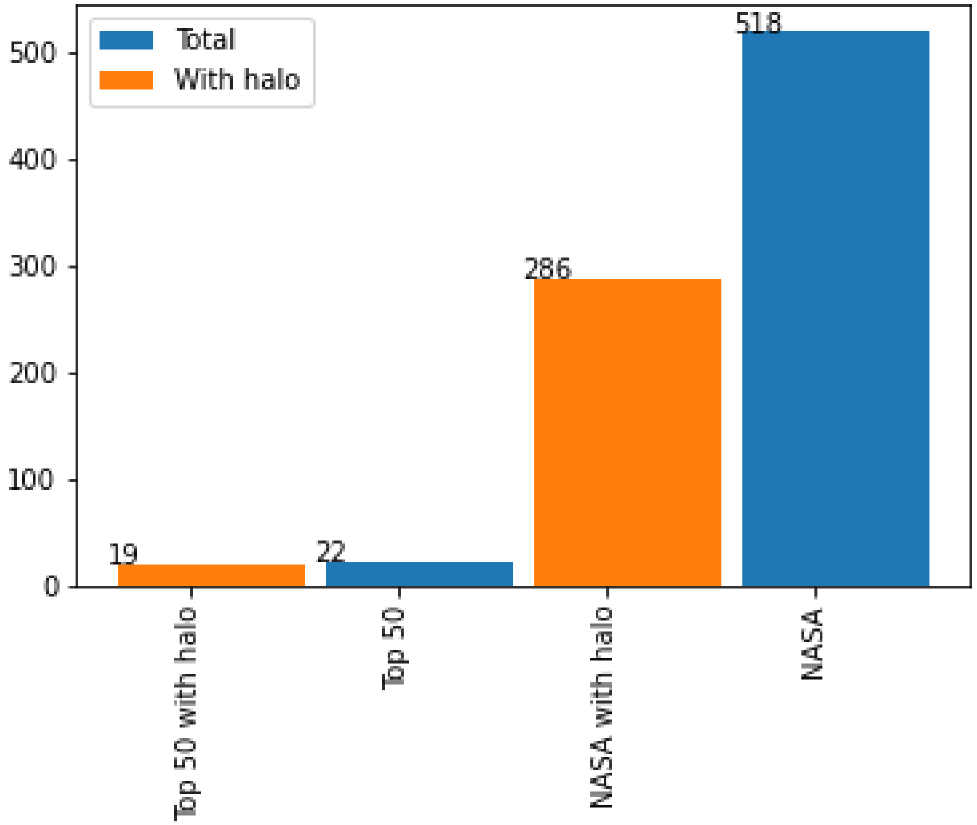
[1410]:

```python
count_all = 0
count_all_halo = 0
for i in range(0,518):
    count_all += 1
    if df4.iat[i,12] == True:
        count_all_halo +=1
print(count_all)
print(count_all_halo)
```

```
518
286
```

[1411]:

```
barWidth = 0.9
bars1 = [22, 518]
bars2 = [19, 286]
bars4 = bars1+bars2

r1 = [2,4]
r2 = [1,3]
r4 = r1+r2

label = ['22', '518','19', '286']

plt.bar(r1, bars1, width = barWidth, label = 'Total')
plt.bar(r2, bars2, width = barWidth, label = 'With halo')

plt.legend()

plt.xticks([r + barWidth for r in range(len(r4))], ['Top 50 with halo','Top␣
 ↪50', 'NASA with halo', 'NASA'], rotation=90)

for i in range(len(r4)):
    plt.text(x = r4[i]-0.5 , y = bars4[i]+0.1, s = label[i], size = 10)

plt.show()
```



[1412]:
```
#Description of Plot:
# The y-axis is the number of solar flares and x-axis has four types:
# Top 50 with halo','Top 50', 'NASA with halo', 'NASA'.
# Organe color means with Halo and blue means without Halo.
# The height represents the number.
```

[1413]:
```
# Yes, flares in the top 50 tend to have Halo CMEs. Based on the graph,
# the difference of height between top50 is much less than all data.
# Also, 19/22 = 0.86... and 286/518 = 0.552...
# Thus, flares in the top 50 tend to have Halo CMEs.
# Thus, we can conclude that as solar flares are stronger, there is more
 ↪possibility that it is with Halo CME.
```

[ ]:

[ ]: