Project1_Dayton

September 27, 2022

[1732]:
```
# Import and grab the data from the website!

import pandas as pd
import requests as r
import numpy as numpy
from bs4 import BeautifulSoup

site = r.get("https://cmsc320.github.io/files/top-50-solar-flares.html")
# parse with BeautifulSoup
soup = BeautifulSoup(site.content, 'html.parser')
prettyParse = soup.prettify()
elements = soup.find_all('tr')

# *** Commented out the prints for output cleanlyness ***
#for i in range (len(elements)):
#    print("*********")
#    print(i)
#    print(elements[i])

# Look for the data we need to grab from the table
#for i in elements[5].findChildren("td"):
#    print(i.get_text())

#for i in range (1, 50):
#    print("****** " + elements[i].findChildren("td")[0].get_text() + " ******")
#    for x in elements[i].findChildren("td"):
#        print(x.get_text())


#dictionary_dataframe = { "Rank" : [], "Classification" : [], "Date" : [],␣
 ↪"Region" : [], "Start" : [], "Maximum" : [], "End" : [], "Recording" : [] }
#for i in range (1, 50):
#    dictionary_dataframe["Rank"].append(elements[i].findChildren("td")[0].
 ↪get_text())
#    dictionary_dataframe["Classification"].append(elements[i].
 ↪findChildren("td")[1].get_text())
```

```
#    dictionary_dataframe["Date"].append(elements[i].findChildren("td")[2].
 ↪get_text())
#    dictionary_dataframe["Region"].append(elements[i].findChildren("td")[3].
 ↪get_text())
#    dictionary_dataframe["Start"].append(elements[i].findChildren("td")[4].
 ↪get_text())
#    dictionary_dataframe["Maximum"].append(elements[i].findChildren("td")[5].
 ↪get_text())
#    dictionary_dataframe["End"].append(elements[i].findChildren("td")[6].
 ↪get_text())
#    dictionary_dataframe["Recording"].append(elements[i].findChildren("td")[7].
 ↪get_text())
# The above code works but it was not what was asked for in the ReadMe

#Get the data from the pretty parse of the webpage and put it into a data frame.
 ↪ Then give it approriate names
dfs = pd.read_html(prettyParse)
dfs[0]


df = dfs[0]
df.rename(columns = {"Unnamed: 0" : "Rank", "Unnamed: 1" : "Classification",␣
 ↪"Unnamed: 2" : "Date", "Unnamed: 7" : "Recording" }, inplace = True)
df
# End of Step 1
```

[1732]:

	Rank	Classification	Date	Region	Start	Maximum	End	\
0	1	X28+	2003/11/04	486	19:29	19:53	20:06	
1	2	X20+	2001/04/02	9393	21:32	21:51	22:03	
2	3	X17.2+	2003/10/28	486	09:51	11:10	11:24	
3	4	X17+	2005/09/07	808	17:17	17:40	18:03	
4	5	X14.4	2001/04/15	9415	13:19	13:50	13:55	
5	6	X10	2003/10/29	486	20:37	20:49	21:01	
6	7	X9.4	1997/11/06	8100	11:49	11:55	12:01	
7	8	X9.3	2017/09/06	2673	11:53	12:02	12:10	
8	9	X9	2006/12/05	930	10:18	10:35	10:45	
9	10	X8.3	2003/11/02	486	17:03	17:25	17:39	
10	11	X8.2	2017/09/10	2673	15:35	16:06	16:31	
11	12	X7.1	2005/01/20	720	06:36	07:01	07:26	
12	13	X6.9	2011/08/09	1263	07:48	08:05	08:08	
13	14	X6.5	2006/12/06	930	18:29	18:47	19:00	
14	15	X6.2	2005/09/09	808	19:13	20:04	20:36	
15	16	X6.2	2001/12/13	9733	14:20	14:30	14:35	
16	17	X5.7	2000/07/14	9077	10:03	10:24	10:43	
17	18	X5.6	2001/04/06	9415	19:10	19:21	19:31	
18	19	X5.4	2012/03/07	1429	00:02	00:24	00:40	

19	20	X5.4	2005/09/08	808	20:52	21:06	21:17
20	21	X5.4	2003/10/23	486	08:19	08:35	08:49
21	22	X5.3	2001/08/25	9591	16:23	16:45	17:04
22	23	X4.9	2014/02/25	1990	00:39	00:49	01:03
23	24	X4.9	1998/08/18	8307	22:10	22:19	22:28
24	25	X4.8	2002/07/23	39	00:18	00:35	00:47
25	26	X4	2000/11/26	9236	16:34	16:48	16:56
26	27	X3.9	2003/11/03	488	09:43	09:55	10:19
27	28	X3.9	1998/08/19	8307	21:35	21:45	21:50
28	29	X3.8	2005/01/17	720	06:59	09:52	10:07
29	30	X3.7	1998/11/22	8384	06:30	06:42	06:49
30	31	X3.6	2005/09/09	808	09:42	09:59	10:08
31	32	X3.6	2004/07/16	649	13:49	13:55	14:01
32	33	X3.6	2003/05/28	365	00:17	00:27	00:39
33	34	X3.4	2006/12/13	930	02:14	02:40	02:57
34	35	X3.4	2001/12/28	9767	20:02	20:45	21:32
35	36	X3.3	2013/11/05	1890	22:07	22:12	22:15
36	37	X3.3	2002/07/20	39	21:04	21:30	21:54
37	38	X3.3	1998/11/28	8395	04:54	05:52	06:13
38	39	X3.2	2013/05/14	1748	00:00	01:11	01:20
39	40	X3.1	2014/10/24	2192	21:07	21:41	22:13
40	41	X3.1	2002/08/24	69	00:49	01:12	01:31
41	42	X3	2002/07/15	30	19:59	20:08	20:14
42	43	X2.8	2013/05/13	1748	15:48	16:05	16:16
43	44	X2.8	2001/12/11	9733	07:58	08:08	08:14
44	45	X2.8	1998/08/18	8307	08:14	08:24	08:32
45	46	X2.7	2015/05/05	2339	22:05	22:11	22:15
46	47	X2.7	2003/11/03	488	01:09	01:30	01:45
47	48	X2.7	1998/05/06	8210	07:58	08:09	08:20
48	49	X2.6	2005/01/15	720	22:25	23:02	23:31
49	50	X2.6	2001/09/24	9632	09:32	10:38	11:09

		Recording
0	Movie	View archive
1	Movie	View archive
2	Movie	View archive
3	Movie	View archive
4	Movie	View archive
5	Movie	View archive
6	Movie	View archive
7	Movie	View archive
8	Movie	View archive
9	Movie	View archive
10	Movie	View archive
11	Movie	View archive
12	Movie	View archive
13	Movie	View archive

```
14  Movie  View archive
15  Movie  View archive
16  Movie  View archive
17  Movie  View archive
18  Movie  View archive
19  Movie  View archive
20  Movie  View archive
21  Movie  View archive
22  Movie  View archive
23         View archive
24  Movie  View archive
25  Movie  View archive
26  Movie  View archive
27         View archive
28  Movie  View archive
29  Movie  View archive
30  Movie  View archive
31  Movie  View archive
32  Movie  View archive
33  Movie  View archive
34  Movie  View archive
35  Movie  View archive
36  Movie  View archive
37  Movie  View archive
38  Movie  View archive
39  Movie  View archive
40  Movie  View archive
41  Movie  View archive
42  Movie  View archive
43  Movie  View archive
44         View archive
45  Movie  View archive
46  Movie  View archive
47  Movie  View archive
48  Movie  View archive
49  Movie  View archive
```

[1733]:
```python
import datetime as dt
# Start of Step 2
# Remove the Recording column of the data frame
df = df.drop(["Recording"], axis = 1)

# Combine the date and time columns
df["Start"] = df["Date"] + " " + df["Start"]
df["Maximum"] = df["Date"] + " " + df["Maximum"]
df["End"] = df["Date"] + " " + df["End"]
```

```
# Using datetime convert the columns with the dates and times in them to a␣
 ↪better format
df["Start"] = pd.to_datetime(df["Start"])
df["Maximum"] = pd.to_datetime(df["Maximum"])
df["End"] = pd.to_datetime(df["End"])

# Get rid of the date column
df = df.drop(["Date"], axis = 1)

df.rename(columns = {"Start" : "Start_Datetime", "Maximum" :␣
 ↪"Maximum_Datetime", "End" : "End_Datetime"}, inplace = True)

# Get rid of the +'s from the classification and if it didn't have a decimal␣
 ↪add one for consistency
df["Classification"] = df["Classification"].apply(lambda x: x.replace("+", ""))
df["Classification"] = df["Classification"].apply(lambda x: x + ".0" if ("." in␣
 ↪x) == False else x)

# replace empty Regions with the mean
df["Region"] = df["Region"].apply(str)
df["Region"] = df["Region"].apply(lambda x: x.replace("-", str(df["Region"].
 ↪mode)))
df["Region"] = df["Region"].apply(int)

df

#End of Step 2
```

[1733]:

	Rank	Classification	Region	Start_Datetime	Maximum_Datetime	\
0	1	X28.0	486	2003-11-04 19:29:00	2003-11-04 19:53:00	
1	2	X20.0	9393	2001-04-02 21:32:00	2001-04-02 21:51:00	
2	3	X17.2	486	2003-10-28 09:51:00	2003-10-28 11:10:00	
3	4	X17.0	808	2005-09-07 17:17:00	2005-09-07 17:40:00	
4	5	X14.4	9415	2001-04-15 13:19:00	2001-04-15 13:50:00	
5	6	X10.0	486	2003-10-29 20:37:00	2003-10-29 20:49:00	
6	7	X9.4	8100	1997-11-06 11:49:00	1997-11-06 11:55:00	
7	8	X9.3	2673	2017-09-06 11:53:00	2017-09-06 12:02:00	
8	9	X9.0	930	2006-12-05 10:18:00	2006-12-05 10:35:00	
9	10	X8.3	486	2003-11-02 17:03:00	2003-11-02 17:25:00	
10	11	X8.2	2673	2017-09-10 15:35:00	2017-09-10 16:06:00	
11	12	X7.1	720	2005-01-20 06:36:00	2005-01-20 07:01:00	
12	13	X6.9	1263	2011-08-09 07:48:00	2011-08-09 08:05:00	
13	14	X6.5	930	2006-12-06 18:29:00	2006-12-06 18:47:00	
14	15	X6.2	808	2005-09-09 19:13:00	2005-09-09 20:04:00	
15	16	X6.2	9733	2001-12-13 14:20:00	2001-12-13 14:30:00	
16	17	X5.7	9077	2000-07-14 10:03:00	2000-07-14 10:24:00	
17	18	X5.6	9415	2001-04-06 19:10:00	2001-04-06 19:21:00	

18	19	X5.4	1429	2012-03-07 00:02:00	2012-03-07 00:24:00
19	20	X5.4	808	2005-09-08 20:52:00	2005-09-08 21:06:00
20	21	X5.4	486	2003-10-23 08:19:00	2003-10-23 08:35:00
21	22	X5.3	9591	2001-08-25 16:23:00	2001-08-25 16:45:00
22	23	X4.9	1990	2014-02-25 00:39:00	2014-02-25 00:49:00
23	24	X4.9	8307	1998-08-18 22:10:00	1998-08-18 22:19:00
24	25	X4.8	39	2002-07-23 00:18:00	2002-07-23 00:35:00
25	26	X4.0	9236	2000-11-26 16:34:00	2000-11-26 16:48:00
26	27	X3.9	488	2003-11-03 09:43:00	2003-11-03 09:55:00
27	28	X3.9	8307	1998-08-19 21:35:00	1998-08-19 21:45:00
28	29	X3.8	720	2005-01-17 06:59:00	2005-01-17 09:52:00
29	30	X3.7	8384	1998-11-22 06:30:00	1998-11-22 06:42:00
30	31	X3.6	808	2005-09-09 09:42:00	2005-09-09 09:59:00
31	32	X3.6	649	2004-07-16 13:49:00	2004-07-16 13:55:00
32	33	X3.6	365	2003-05-28 00:17:00	2003-05-28 00:27:00
33	34	X3.4	930	2006-12-13 02:14:00	2006-12-13 02:40:00
34	35	X3.4	9767	2001-12-28 20:02:00	2001-12-28 20:45:00
35	36	X3.3	1890	2013-11-05 22:07:00	2013-11-05 22:12:00
36	37	X3.3	39	2002-07-20 21:04:00	2002-07-20 21:30:00
37	38	X3.3	8395	1998-11-28 04:54:00	1998-11-28 05:52:00
38	39	X3.2	1748	2013-05-14 00:00:00	2013-05-14 01:11:00
39	40	X3.1	2192	2014-10-24 21:07:00	2014-10-24 21:41:00
40	41	X3.1	69	2002-08-24 00:49:00	2002-08-24 01:12:00
41	42	X3.0	30	2002-07-15 19:59:00	2002-07-15 20:08:00
42	43	X2.8	1748	2013-05-13 15:48:00	2013-05-13 16:05:00
43	44	X2.8	9733	2001-12-11 07:58:00	2001-12-11 08:08:00
44	45	X2.8	8307	1998-08-18 08:14:00	1998-08-18 08:24:00
45	46	X2.7	2339	2015-05-05 22:05:00	2015-05-05 22:11:00
46	47	X2.7	488	2003-11-03 01:09:00	2003-11-03 01:30:00
47	48	X2.7	8210	1998-05-06 07:58:00	1998-05-06 08:09:00
48	49	X2.6	720	2005-01-15 22:25:00	2005-01-15 23:02:00
49	50	X2.6	9632	2001-09-24 09:32:00	2001-09-24 10:38:00

	End_Datetime
0	2003-11-04 20:06:00
1	2001-04-02 22:03:00
2	2003-10-28 11:24:00
3	2005-09-07 18:03:00
4	2001-04-15 13:55:00
5	2003-10-29 21:01:00
6	1997-11-06 12:01:00
7	2017-09-06 12:10:00
8	2006-12-05 10:45:00
9	2003-11-02 17:39:00
10	2017-09-10 16:31:00
11	2005-01-20 07:26:00
12	2011-08-09 08:08:00

```
13 2006-12-06 19:00:00
14 2005-09-09 20:36:00
15 2001-12-13 14:35:00
16 2000-07-14 10:43:00
17 2001-04-06 19:31:00
18 2012-03-07 00:40:00
19 2005-09-08 21:17:00
20 2003-10-23 08:49:00
21 2001-08-25 17:04:00
22 2014-02-25 01:03:00
23 1998-08-18 22:28:00
24 2002-07-23 00:47:00
25 2000-11-26 16:56:00
26 2003-11-03 10:19:00
27 1998-08-19 21:50:00
28 2005-01-17 10:07:00
29 1998-11-22 06:49:00
30 2005-09-09 10:08:00
31 2004-07-16 14:01:00
32 2003-05-28 00:39:00
33 2006-12-13 02:57:00
34 2001-12-28 21:32:00
35 2013-11-05 22:15:00
36 2002-07-20 21:54:00
37 1998-11-28 06:13:00
38 2013-05-14 01:20:00
39 2014-10-24 22:13:00
40 2002-08-24 01:31:00
41 2002-07-15 20:14:00
42 2013-05-13 16:16:00
43 2001-12-11 08:14:00
44 1998-08-18 08:32:00
45 2015-05-05 22:15:00
46 2003-11-03 01:45:00
47 1998-05-06 08:20:00
48 2005-01-15 23:31:00
49 2001-09-24 11:09:00
```

[1734]:
```
# Start of Step 3
# Get the site with request and parse with BeautifulSoup
siteNasa = r.get("https://cmsc320.github.io/files/waves_type2.html")
soupNasa = BeautifulSoup(siteNasa.content, "html.parser")

# Put each line (string) into an element into an array and get rid of the␣
 ↪unneeded lines
nasaLinesList = soupNasa.text.splitlines()
```

```
# remove all the unneeded lines which were pretty visible by printing the text
linesToRemove = list(range(15))
linesToRemove.append(518)
linesToRemove.append(519)
nasaLinesList = numpy.delete(nasaLinesList, linesToRemove)
nasaLinesListSplit = []

# Commented out the prints to cleanlyness
# Look at at the lines
for i in range (len(nasaLinesList)):
    #print("*********")
    #print(i)
    #print(nasaLinesList[i])
    #print("SPLIT:")
    #print(nasaLinesListSplit[i])
    #print(len(nasaLinesListSplit[i]))

    nasaLinesListSplit.append(nasaLinesList[i].split())
    # remove the unneeded info
    if (len(nasaLinesListSplit[i]) > 15) :
        del nasaLinesListSplit[i][15 : len(nasaLinesListSplit[i])]

# The last 2 rows are filled with garbage so delete them
del nasaLinesListSplit[516 : 518]
# print(nasaLinesListSplit)

# Put the 2D list into a DataFrame (All columns are currently strings)
nasaDF = pd.DataFrame(nasaLinesListSplit, columns = ["Start_Date",␣
 ↪"Start_Time", "End_Date", "End_Time", "Start_Frequency",
                                                      "End_Frequency",␣
 ↪"Flare_Location", "Flare_Region", "Flare_Importance",
                                                      "CME_Date", "CME_Time",␣
 ↪"CME_CPA", "CME_Width", "CME_Speed", "PHTX"])
nasaDF

# END OF STEP 3
```

[1734]:

	Start_Date	Start_Time	End_Date	End_Time	Start_Frequency	End_Frequency	\
0	1997/04/01	14:00	04/01	14:15	8000	4000	
1	1997/04/07	14:30	04/07	17:30	11000	1000	
2	1997/05/12	05:15	05/14	16:00	12000	80	
3	1997/05/21	20:20	05/21	22:00	5000	500	
4	1997/09/23	21:53	09/23	22:16	6000	2000	
..	…	…	…	…	…	…	
511	2017/09/04	20:27	09/05	04:54	14000	210	
512	2017/09/06	12:05	09/07	08:00	16000	70	
513	2017/09/10	16:02	09/11	06:50	16000	150	

```
514  2017/09/12      07:38    09/12    07:43           16000          13000
515  2017/09/17      11:45    09/17    12:35           16000            900

    Flare_Location Flare_Region Flare_Importance CME_Date CME_Time CME_CPA  \
0            S25E16         8026             M1.3    04/01    15:18      74
1            S28E19         8027             C6.8    04/07    14:27    Halo
2            N21W08         8038             C1.3    05/12    05:30    Halo
3            N05W12         8040             M1.3    05/21    21:00     263
4            S29E25         8088             C1.4    09/23    22:02     133
..              …            …                …       …        …        …
511          S10W12        12673             M5.5    09/04    20:12    Halo
512          S08W33        12673             X9.3    09/06    12:24    Halo
513          S09W92        -----             X8.3    09/10    16:00    Halo
514          N08E48        12680             C3.0    09/12    08:03     124
515        S08E170        -----             ----    09/17    12:00    Halo

    CME_Width CME_Speed  PHTX
0          79       312  PHTX
1         360       878  PHTX
2         360       464  PHTX
3         165       296  PHTX
4         155       712  PHTX
..        …        …    …
511       360      1418  PHTX
512       360      1571  PHTX
513       360      3163  PHTX
514        96       252  PHTX
515       360      1385  PHTX

[516 rows x 15 columns]
```

[1735]:
```
# Start of Step 4

# Change empty input into NaN. What is considered empty input was gotten from...
# https://web.archive.org/web/20210318011551/http://cdaw.gsfc.nasa.gov/CME_list/
 ↪radio/waves_type2_description.htm
nasaDF["Start_Frequency"] = nasaDF["Start_Frequency"].apply(lambda x: numpy.nan␣
 ↪if x == "????" else x)
nasaDF["End_Frequency"] = nasaDF["End_Frequency"].apply(lambda x: numpy.nan if␣
 ↪x == "????" else x)

nasaDF["Flare_Region"] = nasaDF["Flare_Region"].apply(lambda x: numpy.nan if x␣
 ↪== "-----" else x)
# Some of the Flare_Importance values have a number like "13." I want to make␣
 ↪that "13.0"
nasaDF["Flare_Importance"] = nasaDF["Flare_Importance"].apply(lambda x: str(x)␣
 ↪+ "0" if bool(re.match("(\w\d\.\d+)", str(x))) == False else x)
```

```
nasaDF["Flare_Importance"] = nasaDF["Flare_Importance"].apply(lambda x: numpy.
 ↪nan if x == "----0" else x)
nasaDF["Flare_Importance"] = nasaDF["Flare_Importance"].apply(lambda x: numpy.
 ↪nan if x == "FILA0" else x)
# nasaDF["Flare_Location"].unique() ***** Considering BACK as fine for now *****

nasaDF["CME_CPA"] = nasaDF["CME_CPA"].apply(lambda x: numpy.nan if x == "----"␣
 ↪else x)
nasaDF["CME_Width"] = nasaDF["CME_Width"].apply(lambda x: numpy.nan if x ==␣
 ↪"----" else x)
nasaDF["CME_Width"] = nasaDF["CME_Width"].apply(lambda x: numpy.nan if x ==␣
 ↪"---" else x)
nasaDF["CME_Speed"] = nasaDF["CME_Speed"].apply(lambda x: numpy.nan if x ==␣
 ↪"----" else x)

nasaDF
```

[1735]:

	Start_Date	Start_Time	End_Date	End_Time	Start_Frequency	End_Frequency
0	1997/04/01	14:00	04/01	14:15	8000	4000
1	1997/04/07	14:30	04/07	17:30	11000	1000
2	1997/05/12	05:15	05/14	16:00	12000	80
3	1997/05/21	20:20	05/21	22:00	5000	500
4	1997/09/23	21:53	09/23	22:16	6000	2000
..	…	…	…	…	…	…
511	2017/09/04	20:27	09/05	04:54	14000	210
512	2017/09/06	12:05	09/07	08:00	16000	70
513	2017/09/10	16:02	09/11	06:50	16000	150
514	2017/09/12	07:38	09/12	07:43	16000	13000
515	2017/09/17	11:45	09/17	12:35	16000	900

	Flare_Location	Flare_Region	Flare_Importance	CME_Date	CME_Time	CME_CPA
0	S25E16	8026	M1.3	04/01	15:18	74
1	S28E19	8027	C6.8	04/07	14:27	Halo
2	N21W08	8038	C1.3	05/12	05:30	Halo
3	N05W12	8040	M1.3	05/21	21:00	263
4	S29E25	8088	C1.4	09/23	22:02	133
..	…	…	…	…	…	…
511	S10W12	12673	M5.5	09/04	20:12	Halo
512	S08W33	12673	X9.3	09/06	12:24	Halo
513	S09W92	NaN	X8.3	09/10	16:00	Halo
514	N08E48	12680	C3.0	09/12	08:03	124
515	S08E170	NaN	NaN	09/17	12:00	Halo

	CME_Width	CME_Speed	PHTX
0	79	312	PHTX
1	360	878	PHTX

```
2          360        464  PHTX
3          165        296  PHTX
4          155        712  PHTX
..         …          …    …
511        360       1418  PHTX
512        360       1571  PHTX
513        360       3163  PHTX
514         96        252  PHTX
515        360       1385  PHTX

[516 rows x 15 columns]
```

[1736]:
```
import re # To use Regex
# Combine the date and time columns
nasaDF["Start_Time"] = nasaDF["Start_Date"] + " " + nasaDF["Start_Time"]
nasaDF["End_Time"] = nasaDF["End_Date"] + " " + nasaDF["End_Time"]
nasaDF["CME_Time"] = nasaDF["CME_Date"] + " " + nasaDF["CME_Time"]

# Take Start_Date, since we are not longer using this column, and replace its␣
 ↪data with "year/"
# so we can concat the year onto the front of the dates missing the year.
nasaDF["Start_Date"] = nasaDF["Start_Date"].apply(lambda x: re.split("\/",␣
 ↪x)[0] + "/")
nasaDF["End_Time"] = nasaDF["Start_Date"] + nasaDF["End_Time"]
nasaDF["CME_Time"] = nasaDF["Start_Date"] + nasaDF["CME_Time"]

# Some End Times are at Midnight which is 00:00 not 24:00. When doing this I␣
 ↪need to increment the day by 1.
# To do this I am checking if 24:00 is in the string. Then using regex to grab␣
 ↪the day from the date. Then casting the day to an int.
# Incrementing the day by 1, then formatting it so 2 is 02. From there I am␣
 ↪recasting it into a string and replacing the day with
# the new incremented day. Then I just replace 24:00 with 00:00.
nasaDF["End_Time"] = nasaDF["End_Time"].apply(lambda x: x.replace(re.search("\/
 ↪(\d{2}) ", x).group(1), str("{0:0=2d}".format(int(re.search("\/(\d{2}) ", x).
 ↪group(1)) + 1))) if "24:00" in x else x)
nasaDF["End_Time"] = nasaDF["End_Time"].apply(lambda x: x.replace("24:00", "00:
 ↪00"))

# Turn the time columns into datetime
nasaDF["Start_Time"] = pd.to_datetime(nasaDF["Start_Time"])
nasaDF["End_Time"] = pd.to_datetime(nasaDF["End_Time"])
nasaDF["CME_Time"] = pd.to_datetime(nasaDF["CME_Time"], errors = 'coerce') #␣
 ↪turns invalid times into NaT

# Get rid of the date columns and rename the new dattime columns
```

```
nasaDF = nasaDF.drop(["Start_Date", "End_Date", "CME_Date"], axis = 1)
nasaDF.rename(columns = {"Start_Time" : "Start_Datetime", "End_Time" :␣
 ↪"End_Datetime", "CME_Time" : "CME_Datetime"}, inplace = True)

# Add a column which displays true if it was a halo, otherwise false. Then␣
 ↪replce Halo in the CME_CPA with NaN if it was a halo
HaloMask = nasaDF["CME_CPA"] == "Halo"
nasaDF["Is_Halo"] = HaloMask
nasaDF["CME_CPA"] = nasaDF["CME_CPA"].apply(lambda x: numpy.nan if x == "Halo"␣
 ↪else x)

# The width column indicates if the given value is a lower bound. Create a new␣
 ↪column that indicates if width is given as a lower bound,
# and remove any non-numeric part of the width column.
lowerBoundMask = ">" in nasaDF["CME_Width"]
nasaDF["Is_Lower_Bound"] = lowerBoundMask
nasaDF["CME_Width"] = nasaDF["CME_Width"].apply(lambda x: str(re.
 ↪search("(\d+)", x).group(1)) if ">" in str(x) else x)

nasaDF

# END OF STEP 4
```

[1736]:

	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	
..	…	…	…	…	
511	2017-09-04 20:27:00	2017-09-05 04:54:00	14000	210	
512	2017-09-06 12:05:00	2017-09-07 08:00:00	16000	70	
513	2017-09-10 16:02:00	2017-09-11 06:50:00	16000	150	
514	2017-09-12 07:38:00	2017-09-12 07:43:00	16000	13000	
515	2017-09-17 11:45:00	2017-09-17 12:35:00	16000	900	

	Flare_Location	Flare_Region	Flare_Importance	CME_Datetime	CME_CPA	\
0	S25E16	8026	M1.3	1997-04-01 15:18:00	74	
1	S28E19	8027	C6.8	1997-04-07 14:27:00	NaN	
2	N21W08	8038	C1.3	1997-05-12 05:30:00	NaN	
3	N05W12	8040	M1.3	1997-05-21 21:00:00	263	
4	S29E25	8088	C1.4	1997-09-23 22:02:00	133	
..	…	…	…	…	…	
511	S10W12	12673	M5.5	2017-09-04 20:12:00	NaN	
512	S08W33	12673	X9.3	2017-09-06 12:24:00	NaN	
513	S09W92	NaN	X8.3	2017-09-10 16:00:00	NaN	
514	N08E48	12680	C3.0	2017-09-12 08:03:00	124	

```
515        S08E170          NaN              NaN 2017-09-17 12:00:00     NaN

    CME_Width CME_Speed  PHTX  Is_Halo  Is_Lower_Bound
0          79       312  PHTX    False           False
1         360       878  PHTX     True           False
2         360       464  PHTX     True           False
3         165       296  PHTX    False           False
4         155       712  PHTX    False           False
..        …       …    …       …               …
511       360      1418  PHTX     True           False
512       360      1571  PHTX     True           False
513       360      3163  PHTX     True           False
514        96       252  PHTX    False           False
515       360      1385  PHTX     True           False

[516 rows x 14 columns]
```

[1737]:
```
# START OF PART 2, QUESTION 1. Replication

# We can not replicate the SpaceWeatherLive.com exactly because the Nasa list␣
 ↪does not have all of the same data as the SpaceWeatherLive.com list does.
# For context, Flare_Importance from the Nasa table and Classification from the␣
 ↪SpaceWeatherLive.com table are both measuring Soft X-ray flare size.
# If we look at the two DataFrames (df being the SpaceWeatherLive.com data and␣
 ↪nasaDF being the Nasa data) data, we can see that the Nasa data does not
# share the same top 50 flares. For exmaple SpaceWeatherLive.com, has a flare␣
 ↪which was measured at X17.0 in the year 2005. The Nasa data has a flare
# at X17.0 as well, but that flare was in 2003 meaning they are two seperate␣
 ↪solar flares.



# Organize the nasa dataframe by lare_Importance in decending order.
# We take the number and the letter out of Flare_Importance and put it into␣
 ↪seperate columnns
# From there we sort by the two ranks combined with the letter weighing more to␣
 ↪ensure X's are
# Showing up first before lower letters with high numbers. Then remove the new␣
 ↪columnns once sorted.
nasaDF["sortNum"] = nasaDF["Flare_Importance"].str.extract("(\d+\.\d+)", expand␣
 ↪= False).astype(float)
nasaDF["sortChar"] = nasaDF["Flare_Importance"].str.extract("(\w{1})", expand =␣
 ↪False).astype(str)
nasaDF['Rank'] = nasaDF["sortNum"].rank() + (nasaDF["sortChar"].rank() * 4)
nasaDF = nasaDF.sort_values(by = ["Rank"], ascending = [False])
nasaDF = nasaDF.drop("sortNum", axis=1)
```

```
nasaDF = nasaDF.drop("sortChar", axis=1)
nasaDF = nasaDF.drop("Rank", axis=1)

# How to get the first 50 rows in the Nasa data into a new Dataframe and then␣
 ↪dispay it
nasaTop50 = nasaDF.copy(deep = True)
nasaTop50 = nasaTop50.drop(nasaTop50.tail(466).index)
nasaTop50 = nasaTop50.reset_index(drop=True)
display(nasaTop50)
display(df)

# Write a sentence or two discussing how well you can replicate the␣
 ↪SpaceWeatherLive data from the NASA data.
#␣
 ↪-------------------------------------------------------------------------------------------
# I don't think it would be too difficult to replicate the SpaceWeatherLive.com␣
 ↪data from the Nasa data because the Nasa data does share many of
# the same flares as SpaceWeatherLive.com. It is missing some but we could take␣
 ↪the next closest flare whcih is not too far off in classification
# or importance and call it day. It won't be a perfect copy, but that is okay,␣
 ↪because it will be pretty darn close.

# END OF PART 2, QUESTION 1. Replication
```

	Start_Datetime	End_Datetime	Start_Frequency	End_Frequency	\
0	2003-11-04 20:00:00	2003-11-05 00:00:00	10000	200	
1	2001-04-02 22:05:00	2001-04-03 02:30:00	14000	250	
2	2003-10-28 11:10:00	2003-10-30 00:00:00	14000	40	
3	2001-04-15 14:05:00	2001-04-16 13:00:00	14000	40	
4	2003-10-29 20:55:00	2003-10-30 00:00:00	11000	500	
5	1997-11-06 12:20:00	1997-11-07 08:30:00	14000	100	
6	2017-09-06 12:05:00	2017-09-07 08:00:00	16000	70	
7	2006-12-05 10:50:00	2006-12-05 20:00:00	14000	250	
8	2017-09-10 16:02:00	2017-09-11 06:50:00	16000	150	
9	2003-11-02 17:30:00	2003-11-03 01:00:00	12000	250	
10	2005-01-20 07:15:00	2005-01-20 16:30:00	14000	25	
11	2011-08-09 08:20:00	2011-08-09 08:35:00	16000	4000	
12	2006-12-06 19:00:00	2006-12-09 00:00:00	16000	30	
13	2005-09-09 19:45:00	2005-09-09 22:00:00	10000	50	
14	2000-07-14 10:30:00	2000-07-15 14:30:00	14000	80	
15	2001-04-06 19:35:00	2001-04-07 01:50:00	14000	230	
16	2012-03-07 01:00:00	2012-03-08 19:00:00	16000	30	
17	2001-08-25 16:50:00	2001-08-25 23:00:00	8000	170	
18	2014-02-25 00:56:00	2014-02-25 11:28:00	14000	100	
19	2002-07-23 00:50:00	2002-07-23 04:00:00	11000	400	
20	2000-11-26 17:00:00	2000-11-26 17:15:00	14000	7000	
21	2003-11-03 10:00:00	2003-11-03 12:30:00	6000	400	

```
22 2005-01-17 10:00:00 2005-01-17 10:35:00           6100          1500
23 2003-05-28 01:00:00 2003-05-29 00:30:00           1000           200
24 2001-12-28 20:35:00 2001-12-29 03:00:00          14000           350
25 2006-12-13 02:45:00 2006-12-13 10:40:00          12000           150
26 2002-07-20 21:30:00 2002-07-20 22:20:00          10000          2000
27 2013-05-14 01:16:00 2013-05-14 08:20:00          16000           240
28 2002-08-24 01:45:00 2002-08-24 03:25:00           5000           400
29 2013-05-13 16:15:00 2013-05-13 19:10:00          16000           300
30 2015-05-05 22:24:00 2015-05-05 23:14:00          14000           500
31 1998-05-06 08:25:00 1998-05-06 08:35:00          14000          5000
32 2003-11-03 01:15:00 2003-11-03 01:25:00           3000          1500
33 2005-01-15 23:00:00 2005-01-17 00:00:00           3000            40
34 2001-09-24 10:45:00 2001-09-25 20:00:00           7000            30
35 1997-11-27 13:30:00 1997-11-27 14:00:00          14000          7000
36 2004-11-10 02:25:00 2004-11-10 03:40:00          14000          1000
37 2000-06-06 15:20:00 2000-06-08 09:00:00          14000            40
38 2000-11-24 15:25:00 2000-11-24 22:00:00          14000           200
39 2001-04-10 05:24:00 2001-04-11 00:00:00          14000           100
40 2011-02-15 02:10:00 2011-02-15 07:00:00          16000           400
41 2005-09-10 21:45:00 2005-09-11 01:00:00          14000           200
42 1997-11-04 06:00:00 1997-11-05 04:30:00          14000           100
43 2011-09-06 22:30:00 2011-09-07 15:40:00          16000           150
44 2013-10-25 15:08:00 2013-10-25 22:32:00          16000           200
45 2004-11-07 16:25:00 2004-11-08 20:00:00          14000            60
46 2000-11-24 05:10:00 2000-11-24 15:00:00          14000           100
47 2001-04-12 10:20:00 2001-04-12 10:40:00          14000          7000
48 2005-01-17 09:25:00 2005-01-17 16:00:00          14000            30
49 2000-11-25 19:00:00 2000-11-25 19:35:00           6000          2000

   Flare_Location Flare_Region Flare_Importance        CME_Datetime CME_CPA  \
0          S19W83        10486             X28.0 2003-11-04 19:54:00     NaN
1          N19W72         9393             X20.0 2001-04-02 22:06:00     261
2          S16E08        10486             X17.0 2003-10-28 11:30:00     NaN
3          S20W85         9415             X14.0 2001-04-15 14:06:00     245
4          S15W02        10486             X10.0 2003-10-29 20:54:00     NaN
5          S18W63         8100              X9.4 1997-11-06 12:10:00     NaN
6          S08W33        12673              X9.3 2017-09-06 12:24:00     NaN
7          S07E68        10930              X9.0                 NaT     NaN
8          S09W92          NaN              X8.3 2017-09-10 16:00:00     NaN
9          S14W56        10486              X8.3 2003-11-02 17:30:00     NaN
10         N14W61        10720              X7.1 2005-01-20 06:54:00     NaN
11         N17W69        11263              X6.9 2011-08-09 08:12:00     NaN
12         S05E64        10930              X6.5                 NaT     NaN
13         S12E67        10808              X6.2 2005-09-09 19:48:00     NaN
14         N22W07         9077              X5.7 2000-07-14 10:54:00     NaN
15         S21E31         9415              X5.6 2001-04-06 19:30:00     NaN
16         N17E27        11429              X5.4 2012-03-07 00:24:00     NaN
17         S17E34         9591              X5.3 2001-08-25 16:50:00     NaN
```

18	S12E82	11990	X4.9	2014-02-25 01:25:00	NaN
19	S13E72	10039	X4.8	2002-07-23 00:42:00	NaN
20	N18W38	9236	X4.0	2000-11-26 17:06:00	NaN
21	N08W77	10488	X3.9	2003-11-03 10:06:00	293
22	N15W25	10720	X3.8	2005-01-17 09:54:00	NaN
23	S07W20	10365	X3.6	2003-05-28 00:50:00	NaN
24	S26E90	9756	X3.4	2001-12-28 20:30:00	NaN
25	S06W23	10930	X3.4	2006-12-13 02:54:00	NaN
26	S13E90	10039	X3.3	2002-07-20 22:06:00	NaN
27	N08E77	11748	X3.2	2013-05-14 01:25:00	NaN
28	S02W81	10069	X3.1	2002-08-24 01:27:00	NaN
29	N11E85	11748	X2.8	2013-05-13 16:07:00	NaN
30	N15E79	12339	X2.7	2015-05-05 22:24:00	NaN
31	S11W65	8210	X2.7	1998-05-06 08:29:00	309
32	N10W83	10488	X2.7	2003-11-03 01:59:00	304
33	N15W05	10720	X2.6	2005-01-15 23:06:00	NaN
34	S16E23	9632	X2.6	2001-09-24 10:30:00	NaN
35	N17E63	8113	X2.6	1997-11-27 13:56:00	98
36	N09W49	10696	X2.5	2004-11-10 02:26:00	NaN
37	N20E18	9026	X2.3	2000-06-06 15:54:00	NaN
38	N22W07	9236	X2.3	2000-11-24 15:30:00	NaN
39	S23W09	9415	X2.3	2001-04-10 05:30:00	NaN
40	S20W12	11158	X2.2	2011-02-15 02:24:00	NaN
41	S13E47	10808	X2.1	2005-09-10 21:52:00	NaN
42	S14W33	8100	X2.1	1997-11-04 06:10:00	NaN
43	N14W18	11283	X2.1	2011-09-06 23:05:00	NaN
44	S06E69	11882	X2.1	2013-10-25 15:12:00	NaN
45	N09W17	10696	X2.0	2004-11-07 16:54:00	NaN
46	N20W05	9236	X2.0	2000-11-24 05:30:00	NaN
47	S19W43	9415	X2.0	2001-04-12 10:31:00	NaN
48	N15W25	10720	X2.0	2005-01-17 09:30:00	NaN
49	N20W23	9236	X1.9	2000-11-25 19:31:00	NaN

	CME_Width	CME_Speed	PHTX	Is_Halo	Is_Lower_Bound
0	360	2657	PHTX	True	False
1	244	2505	PHTX	False	False
2	360	2459	PHTX	True	False
3	167	1199	PHTX	False	False
4	360	2029	PHTX	True	False
5	360	1556	PHTX	True	False
6	360	1571	PHTX	True	False
7	NaN	NaN	PHTX	False	False
8	360	3163	PHTX	True	False
9	360	2598	PHTX	True	False
10	360	882	PHTX	True	False
11	360	1610	PHTX	True	False
12	NaN	NaN	PHTX	False	False
13	360	2257	PHTX	True	False

```
14       360      1674  PHTX     True           False
15       360      1270  PHTX     True           False
16       360      2684  PHTX     True           False
17       360      1433  PHTX     True           False
18       360      2147  PHTX     True           False
19       360      2285  PHTX     True           False
20       360       980  PHTX     True           False
21       103      1420  PHTX    False           False
22       360      2547  PHTX     True           False
23       360      1366  PHTX     True           False
24       360      2216  PHTX     True           False
25       360      1774  PHTX     True           False
26       360      1941  PHTX     True           False
27       360      2625  PHTX     True           False
28       360      1913  PHTX     True           False
29       360      1850  PHTX     True           False
30       360       715  PHTX     True           False
31       190      1099  PHTX    False           False
32        65       827  PHTX    False           False
33       360      2861  PHTX     True           False
34       360      2402  PHTX     True           False
35        91       441  PHTX    False           False
36       360      3387  PHTX     True           False
37       360      1119  PHTX     True           False
38       360      1245  PHTX     True           False
39       360      2411  PHTX     True           False
40       360       669  PHTX     True           False
41       360      1893  PHTX     True           False
42       360       785  PHTX     True           False
43       360       575  PHTX     True           False
44       360      1081  PHTX     True           False
45       360      1759  PHTX     True           False
46       360      1289  PHTX     True           False
47       360      1184  PHTX     True           False
48       360      2094  PHTX     True           False
49       360       671  PHTX     True           False

    Rank Classification  Region      Start_Datetime    Maximum_Datetime  \
0      1           X28.0     486 2003-11-04 19:29:00 2003-11-04 19:53:00
1      2           X20.0    9393 2001-04-02 21:32:00 2001-04-02 21:51:00
2      3           X17.2     486 2003-10-28 09:51:00 2003-10-28 11:10:00
3      4           X17.0     808 2005-09-07 17:17:00 2005-09-07 17:40:00
4      5           X14.4    9415 2001-04-15 13:19:00 2001-04-15 13:50:00
5      6           X10.0     486 2003-10-29 20:37:00 2003-10-29 20:49:00
6      7            X9.4    8100 1997-11-06 11:49:00 1997-11-06 11:55:00
7      8            X9.3    2673 2017-09-06 11:53:00 2017-09-06 12:02:00
8      9            X9.0     930 2006-12-05 10:18:00 2006-12-05 10:35:00
9     10            X8.3     486 2003-11-02 17:03:00 2003-11-02 17:25:00
```

10	11	X8.2	2673	2017-09-10 15:35:00	2017-09-10 16:06:00
11	12	X7.1	720	2005-01-20 06:36:00	2005-01-20 07:01:00
12	13	X6.9	1263	2011-08-09 07:48:00	2011-08-09 08:05:00
13	14	X6.5	930	2006-12-06 18:29:00	2006-12-06 18:47:00
14	15	X6.2	808	2005-09-09 19:13:00	2005-09-09 20:04:00
15	16	X6.2	9733	2001-12-13 14:20:00	2001-12-13 14:30:00
16	17	X5.7	9077	2000-07-14 10:03:00	2000-07-14 10:24:00
17	18	X5.6	9415	2001-04-06 19:10:00	2001-04-06 19:21:00
18	19	X5.4	1429	2012-03-07 00:02:00	2012-03-07 00:24:00
19	20	X5.4	808	2005-09-08 20:52:00	2005-09-08 21:06:00
20	21	X5.4	486	2003-10-23 08:19:00	2003-10-23 08:35:00
21	22	X5.3	9591	2001-08-25 16:23:00	2001-08-25 16:45:00
22	23	X4.9	1990	2014-02-25 00:39:00	2014-02-25 00:49:00
23	24	X4.9	8307	1998-08-18 22:10:00	1998-08-18 22:19:00
24	25	X4.8	39	2002-07-23 00:18:00	2002-07-23 00:35:00
25	26	X4.0	9236	2000-11-26 16:34:00	2000-11-26 16:48:00
26	27	X3.9	488	2003-11-03 09:43:00	2003-11-03 09:55:00
27	28	X3.9	8307	1998-08-19 21:35:00	1998-08-19 21:45:00
28	29	X3.8	720	2005-01-17 06:59:00	2005-01-17 09:52:00
29	30	X3.7	8384	1998-11-22 06:30:00	1998-11-22 06:42:00
30	31	X3.6	808	2005-09-09 09:42:00	2005-09-09 09:59:00
31	32	X3.6	649	2004-07-16 13:49:00	2004-07-16 13:55:00
32	33	X3.6	365	2003-05-28 00:17:00	2003-05-28 00:27:00
33	34	X3.4	930	2006-12-13 02:14:00	2006-12-13 02:40:00
34	35	X3.4	9767	2001-12-28 20:02:00	2001-12-28 20:45:00
35	36	X3.3	1890	2013-11-05 22:07:00	2013-11-05 22:12:00
36	37	X3.3	39	2002-07-20 21:04:00	2002-07-20 21:30:00
37	38	X3.3	8395	1998-11-28 04:54:00	1998-11-28 05:52:00
38	39	X3.2	1748	2013-05-14 00:00:00	2013-05-14 01:11:00
39	40	X3.1	2192	2014-10-24 21:07:00	2014-10-24 21:41:00
40	41	X3.1	69	2002-08-24 00:49:00	2002-08-24 01:12:00
41	42	X3.0	30	2002-07-15 19:59:00	2002-07-15 20:08:00
42	43	X2.8	1748	2013-05-13 15:48:00	2013-05-13 16:05:00
43	44	X2.8	9733	2001-12-11 07:58:00	2001-12-11 08:08:00
44	45	X2.8	8307	1998-08-18 08:14:00	1998-08-18 08:24:00
45	46	X2.7	2339	2015-05-05 22:05:00	2015-05-05 22:11:00
46	47	X2.7	488	2003-11-03 01:09:00	2003-11-03 01:30:00
47	48	X2.7	8210	1998-05-06 07:58:00	1998-05-06 08:09:00
48	49	X2.6	720	2005-01-15 22:25:00	2005-01-15 23:02:00
49	50	X2.6	9632	2001-09-24 09:32:00	2001-09-24 10:38:00

	End_Datetime
0	2003-11-04 20:06:00
1	2001-04-02 22:03:00
2	2003-10-28 11:24:00
3	2005-09-07 18:03:00
4	2001-04-15 13:55:00
5	2003-10-29 21:01:00

```
6  1997-11-06 12:01:00
7  2017-09-06 12:10:00
8  2006-12-05 10:45:00
9  2003-11-02 17:39:00
10 2017-09-10 16:31:00
11 2005-01-20 07:26:00
12 2011-08-09 08:08:00
13 2006-12-06 19:00:00
14 2005-09-09 20:36:00
15 2001-12-13 14:35:00
16 2000-07-14 10:43:00
17 2001-04-06 19:31:00
18 2012-03-07 00:40:00
19 2005-09-08 21:17:00
20 2003-10-23 08:49:00
21 2001-08-25 17:04:00
22 2014-02-25 01:03:00
23 1998-08-18 22:28:00
24 2002-07-23 00:47:00
25 2000-11-26 16:56:00
26 2003-11-03 10:19:00
27 1998-08-19 21:50:00
28 2005-01-17 10:07:00
29 1998-11-22 06:49:00
30 2005-09-09 10:08:00
31 2004-07-16 14:01:00
32 2003-05-28 00:39:00
33 2006-12-13 02:57:00
34 2001-12-28 21:32:00
35 2013-11-05 22:15:00
36 2002-07-20 21:54:00
37 1998-11-28 06:13:00
38 2013-05-14 01:20:00
39 2014-10-24 22:13:00
40 2002-08-24 01:31:00
41 2002-07-15 20:14:00
42 2013-05-13 16:16:00
43 2001-12-11 08:14:00
44 1998-08-18 08:32:00
45 2015-05-05 22:15:00
46 2003-11-03 01:45:00
47 1998-05-06 08:20:00
48 2005-01-15 23:31:00
49 2001-09-24 11:09:00
```

[1738]:
```
# START OF PART 2, QUESTION 2. Integration
```

```
# For each of the top 50 solar flares in the SpaceWeatherLive data, find the best matching row from the NASA data.
# Here, you have to decide for yourself how you determine what "best matching" means in this context (you will have to justify your approach!)
# Multiple flares may match to the same row from the NASA data, depending on your chosen method, you will be expected to notice this if it occurs.
# In your submission, include an explanation of how you are defining best matching rows across the two datasets in addition to the code used to find the best matches.
# Finally, use your function to add a new column to the NASA dataset indicating its rank according to SpaceWeatherLive, if it appears in that dataset.
# If more than one SpaceWeatherLive entry "best matches", choose one and explain how you chose.
#
# ------------------------------------------------------------------------------------------

# There are many ways to determine which flares from the Nasa data is "best matching" with the SpaceWeatherLive.com data.
# Idealling I would rank the Nasa data in the same way the SpaceWeatherLive.com data was ranked, by Classification or importantance. Doing that
# would allow me have no flares out of order by the SpaceWeatherLive.com standards.
# However, I felt this was not exactly what you were asking for in the prompt...

# So instead, I am going to determine "best matching" as meaning a flare that happened on the same day with the same classification or importance.
# This is because the most unique data set is what day this flare actually occured. The specfic times of day are too specfic to get any useable match,
# but the day itself is still broad enough to get matches, but unique enough to not get too many false positives. When coupled with classifcation there were no false positives.
# Since there will be some that don't have a direct match, I put those in the next available ranking from highest classification to least.
# As in the highest classifcation flare that was a non-match got placed in the highest un-matched rank.
# That is why some flares which based on classification only, would get placed way lower, got instead got ranked high up.

# Extract the start date and a truncated version of the classifcation from both DataFrames.
nasaTop50["Start_Date"] = nasaTop50["Start_Datetime"].apply(lambda x: str(x.date()))
nasaTop50["classNoDecimal"] = nasaTop50["Flare_Importance"].apply(lambda x: re.split("\.", str(x))[0])
nasaTop50["Start_Date_Class"] = nasaTop50["Start_Date"] + " " + nasaTop50["classNoDecimal"]
```

```
df["Start_Date"] = df["Start_Datetime"].apply(lambda x: str(x.date()))
df["classNoDecimal"] = df["Classification"].apply(lambda x: re.split("\.",␣
 ↪str(x))[0])
df["Start_Date_Class"] = df["Start_Date"] + " " + df["classNoDecimal"]

spaceWeatherStartDateClass = df["Start_Date_Class"]

# Check the SpaceWeatherLive.com Start_Date_Class column and compare it to the␣
 ↪Nasa Start_Date_Class column.
def rankByBestMatch(FlareInfo, series):
  for index, element in enumerate(spaceWeatherStartDateClass):
    if element == FlareInfo:
        return index + 1

# If there was no match, fill in the remaining data with the un-matched flare␣
 ↪with the highest classifcation or importance
def fillInRemainingRanks(series):
    newSeries = series.copy(deep = True)
    for index, element in enumerate(newSeries):
        if numpy.isnan(element):
            num = 1
            while (num < 51) :
                if not (num in newSeries.unique()):
                    newSeries[index] = num
                    break
                num = num + 1
    return newSeries

# Run the functions against our data to rank based on "best match"
nasaTop50["Start_Date_Class"] = nasaTop50["Start_Date_Class"].apply(lambda x:␣
 ↪rankByBestMatch(x, nasaTop50["Start_Date_Class"]))
nasaTop50["Start_Date_Class"] =␣
 ↪fillInRemainingRanks(nasaTop50["Start_Date_Class"])

# Clean up the DataFrame by sorting based on the new rank and removing the␣
 ↪unused columns
nasaTop50 = nasaTop50.sort_values(by = ["Start_Date_Class"], ascending = [True])
nasaTop50.rename(columns = {"Start_Date_Class" : "Rank"}, inplace = True)
nasaTop50 = nasaTop50.drop("Start_Date", axis=1)
nasaTop50 = nasaTop50.drop("classNoDecimal", axis=1)

display(nasaTop50)


# END OF PART 2, QUESTION 2. Integration
```

```
        Start_Datetime        End_Datetime Start_Frequency End_Frequency  \
```

0	2003-11-04 20:00:00	2003-11-05 00:00:00	10000	200
1	2001-04-02 22:05:00	2001-04-03 02:30:00	14000	250
2	2003-10-28 11:10:00	2003-10-30 00:00:00	14000	40
35	1997-11-27 13:30:00	1997-11-27 14:00:00	14000	7000
3	2001-04-15 14:05:00	2001-04-16 13:00:00	14000	40
4	2003-10-29 20:55:00	2003-10-30 00:00:00	11000	500
5	1997-11-06 12:20:00	1997-11-07 08:30:00	14000	100
6	2017-09-06 12:05:00	2017-09-07 08:00:00	16000	70
7	2006-12-05 10:50:00	2006-12-05 20:00:00	14000	250
9	2003-11-02 17:30:00	2003-11-03 01:00:00	12000	250
8	2017-09-10 16:02:00	2017-09-11 06:50:00	16000	150
10	2005-01-20 07:15:00	2005-01-20 16:30:00	14000	25
11	2011-08-09 08:20:00	2011-08-09 08:35:00	16000	4000
12	2006-12-06 19:00:00	2006-12-09 00:00:00	16000	30
13	2005-09-09 19:45:00	2005-09-09 22:00:00	10000	50
36	2004-11-10 02:25:00	2004-11-10 03:40:00	14000	1000
14	2000-07-14 10:30:00	2000-07-15 14:30:00	14000	80
15	2001-04-06 19:35:00	2001-04-07 01:50:00	14000	230
16	2012-03-07 01:00:00	2012-03-08 19:00:00	16000	30
37	2000-06-06 15:20:00	2000-06-08 09:00:00	14000	40
38	2000-11-24 15:25:00	2000-11-24 22:00:00	14000	200
17	2001-08-25 16:50:00	2001-08-25 23:00:00	8000	170
18	2014-02-25 00:56:00	2014-02-25 11:28:00	14000	100
39	2001-04-10 05:24:00	2001-04-11 00:00:00	14000	100
19	2002-07-23 00:50:00	2002-07-23 04:00:00	11000	400
20	2000-11-26 17:00:00	2000-11-26 17:15:00	14000	7000
21	2003-11-03 10:00:00	2003-11-03 12:30:00	6000	400
40	2011-02-15 02:10:00	2011-02-15 07:00:00	16000	400
22	2005-01-17 10:00:00	2005-01-17 10:35:00	6100	1500
41	2005-09-10 21:45:00	2005-09-11 01:00:00	14000	200
42	1997-11-04 06:00:00	1997-11-05 04:30:00	14000	100
43	2011-09-06 22:30:00	2011-09-07 15:40:00	16000	150
23	2003-05-28 01:00:00	2003-05-29 00:30:00	1000	200
25	2006-12-13 02:45:00	2006-12-13 10:40:00	12000	150
24	2001-12-28 20:35:00	2001-12-29 03:00:00	14000	350
44	2013-10-25 15:08:00	2013-10-25 22:32:00	16000	200
26	2002-07-20 21:30:00	2002-07-20 22:20:00	10000	2000
45	2004-11-07 16:25:00	2004-11-08 20:00:00	14000	60
27	2013-05-14 01:16:00	2013-05-14 08:20:00	16000	240
46	2000-11-24 05:10:00	2000-11-24 15:00:00	14000	100
28	2002-08-24 01:45:00	2002-08-24 03:25:00	5000	400
47	2001-04-12 10:20:00	2001-04-12 10:40:00	14000	7000
29	2013-05-13 16:15:00	2013-05-13 19:10:00	16000	300
48	2005-01-17 09:25:00	2005-01-17 16:00:00	14000	30
49	2000-11-25 19:00:00	2000-11-25 19:35:00	6000	2000
30	2015-05-05 22:24:00	2015-05-05 23:14:00	14000	500
32	2003-11-03 01:15:00	2003-11-03 01:25:00	3000	1500
31	1998-05-06 08:25:00	1998-05-06 08:35:00	14000	5000

```
33 2005-01-15 23:00:00 2005-01-17 00:00:00           3000            40
34 2001-09-24 10:45:00 2001-09-25 20:00:00           7000            30

   Flare_Location Flare_Region Flare_Importance        CME_Datetime CME_CPA  \
0          S19W83        10486            X28.0 2003-11-04 19:54:00     NaN
1          N19W72         9393            X20.0 2001-04-02 22:06:00     261
2          S16E08        10486            X17.0 2003-10-28 11:30:00     NaN
35         N17E63         8113             X2.6 1997-11-27 13:56:00      98
3          S20W85         9415            X14.0 2001-04-15 14:06:00     245
4          S15W02        10486            X10.0 2003-10-29 20:54:00     NaN
5          S18W63         8100             X9.4 1997-11-06 12:10:00     NaN
6          S08W33        12673             X9.3 2017-09-06 12:24:00     NaN
7          S07E68        10930             X9.0                 NaT     NaN
9          S14W56        10486             X8.3 2003-11-02 17:30:00     NaN
8          S09W92          NaN             X8.3 2017-09-10 16:00:00     NaN
10         N14W61        10720             X7.1 2005-01-20 06:54:00     NaN
11         N17W69        11263             X6.9 2011-08-09 08:12:00     NaN
12         S05E64        10930             X6.5                 NaT     NaN
13         S12E67        10808             X6.2 2005-09-09 19:48:00     NaN
36         N09W49        10696             X2.5 2004-11-10 02:26:00     NaN
14         N22W07         9077             X5.7 2000-07-14 10:54:00     NaN
15         S21E31         9415             X5.6 2001-04-06 19:30:00     NaN
16         N17E27        11429             X5.4 2012-03-07 00:24:00     NaN
37         N20E18         9026             X2.3 2000-06-06 15:54:00     NaN
38         N22W07         9236             X2.3 2000-11-24 15:30:00     NaN
17         S17E34         9591             X5.3 2001-08-25 16:50:00     NaN
18         S12E82        11990             X4.9 2014-02-25 01:25:00     NaN
39         S23W09         9415             X2.3 2001-04-10 05:30:00     NaN
19         S13E72        10039             X4.8 2002-07-23 00:42:00     NaN
20         N18W38         9236             X4.0 2000-11-26 17:06:00     NaN
21         N08W77        10488             X3.9 2003-11-03 10:06:00     293
40         S20W12        11158             X2.2 2011-02-15 02:24:00     NaN
22         N15W25        10720             X3.8 2005-01-17 09:54:00     NaN
41         S13E47        10808             X2.1 2005-09-10 21:52:00     NaN
42         S14W33         8100             X2.1 1997-11-04 06:10:00     NaN
43         N14W18        11283             X2.1 2011-09-06 23:05:00     NaN
23         S07W20        10365             X3.6 2003-05-28 00:50:00     NaN
25         S06W23        10930             X3.4 2006-12-13 02:54:00     NaN
24         S26E90         9756             X3.4 2001-12-28 20:30:00     NaN
44         S06E69        11882             X2.1 2013-10-25 15:12:00     NaN
26         S13E90        10039             X3.3 2002-07-20 22:06:00     NaN
45         N09W17        10696             X2.0 2004-11-07 16:54:00     NaN
27         N08E77        11748             X3.2 2013-05-14 01:25:00     NaN
46         N20W05         9236             X2.0 2000-11-24 05:30:00     NaN
28         S02W81        10069             X3.1 2002-08-24 01:27:00     NaN
47         S19W43         9415             X2.0 2001-04-12 10:31:00     NaN
29         N11E85        11748             X2.8 2013-05-13 16:07:00     NaN
48         N15W25        10720             X2.0 2005-01-17 09:30:00     NaN
```

49	N20W23	9236	X1.9	2000-11-25 19:31:00	NaN
30	N15E79	12339	X2.7	2015-05-05 22:24:00	NaN
32	N10W83	10488	X2.7	2003-11-03 01:59:00	304
31	S11W65	8210	X2.7	1998-05-06 08:29:00	309
33	N15W05	10720	X2.6	2005-01-15 23:06:00	NaN
34	S16E23	9632	X2.6	2001-09-24 10:30:00	NaN

	CME_Width	CME_Speed	PHTX	Is_Halo	Is_Lower_Bound	Rank
0	360	2657	PHTX	True	False	1.0
1	244	2505	PHTX	False	False	2.0
2	360	2459	PHTX	True	False	3.0
35	91	441	PHTX	False	False	4.0
3	167	1199	PHTX	False	False	5.0
4	360	2029	PHTX	True	False	6.0
5	360	1556	PHTX	True	False	7.0
6	360	1571	PHTX	True	False	8.0
7	NaN	NaN	PHTX	False	False	9.0
9	360	2598	PHTX	True	False	10.0
8	360	3163	PHTX	True	False	11.0
10	360	882	PHTX	True	False	12.0
11	360	1610	PHTX	True	False	13.0
12	NaN	NaN	PHTX	False	False	14.0
13	360	2257	PHTX	True	False	15.0
36	360	3387	PHTX	True	False	16.0
14	360	1674	PHTX	True	False	17.0
15	360	1270	PHTX	True	False	18.0
16	360	2684	PHTX	True	False	19.0
37	360	1119	PHTX	True	False	20.0
38	360	1245	PHTX	True	False	21.0
17	360	1433	PHTX	True	False	22.0
18	360	2147	PHTX	True	False	23.0
39	360	2411	PHTX	True	False	24.0
19	360	2285	PHTX	True	False	25.0
20	360	980	PHTX	True	False	26.0
21	103	1420	PHTX	False	False	27.0
40	360	669	PHTX	True	False	28.0
22	360	2547	PHTX	True	False	29.0
41	360	1893	PHTX	True	False	30.0
42	360	785	PHTX	True	False	31.0
43	360	575	PHTX	True	False	32.0
23	360	1366	PHTX	True	False	33.0
25	360	1774	PHTX	True	False	34.0
24	360	2216	PHTX	True	False	35.0
44	360	1081	PHTX	True	False	36.0
26	360	1941	PHTX	True	False	37.0
45	360	1759	PHTX	True	False	38.0
27	360	2625	PHTX	True	False	39.0
46	360	1289	PHTX	True	False	40.0

```
28       360      1913  PHTX     True           False  41.0
47       360      1184  PHTX     True           False  42.0
29       360      1850  PHTX     True           False  43.0
48       360      2094  PHTX     True           False  44.0
49       360       671  PHTX     True           False  45.0
30       360       715  PHTX     True           False  46.0
32        65       827  PHTX    False           False  47.0
31       190      1099  PHTX    False           False  48.0
33       360      2861  PHTX     True           False  49.0
34       360      2402  PHTX     True           False  50.0
```

[1739]:
```
# Start of Part 2, Question 3: Analysis

# The visualization I decided upon was the top 50 flares by classifcation over␣
 ↪CME speed.
# This was to see if there was any correlation between CME speed and having a␣
 ↪high classification.

# Setup the data in the correct format
nasaTop50["X-Classification"] = nasaDF["Flare_Importance"].str.extract("(\d+\.
 ↪\d+)", expand = False).astype(float)
nasaTop50["CME_Speed"] = nasaTop50["CME_Speed"].apply(lambda x: float(x))
# display(nasaTop50["CME_Speed"].describe())

# Draw the scatter plot
nasaTop50.plot.scatter(x = "CME_Speed", y = "X-Classification", alpha = 0.65, s␣
 ↪= 50)

# This scatterplot has the top 50 Nasa flare's X classification (number)␣
 ↪plotted over their CME Speed.
# The alpha has been lowered by 35% and the scale of each plot point has been␣
 ↪increased to see overlap better.

# In terms of correliation there is no real positive or negative correlation.␣
 ↪As in there is no correlation between CME Speed
# and how high a flare will be classified. This is based on how scattered the␣
 ↪data points are. If there was a distinct correlation there
# would be easily recongnizable visual lines we could draw through the graph␣
 ↪where plots appear to be.


# End of Part 2, Question 3: Analysis
```

```
count      48.000000
mean     1731.625000
std       736.920465
min       441.000000
```

```
25%      1167.750000
50%      1716.500000
75%      2314.250000
max      3387.000000
Name: CME_Speed, dtype: float64
```

```
[1739]: <AxesSubplot:xlabel='CME_Speed', ylabel='X-Classification'>
```