Part 1: Data scraping and preparation

Step 1: Scrape your competitor's data

In [2]:
```
# Get necessary Python packages
import requests
from bs4 import BeautifulSoup
import pandas as pd
import numpy as np
```

In [3]:
```
# Read from SpaceWeatherLive.com
headers = {
    'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
    'From': 'pleaseletmein@gmail.com'
}

r = requests.get("https://www.spaceweatherlive.com/en/solar-activity/top-50-sol

# Check status code
if r.status_code != 200:
  print("Request failed from the provided website.")
else:
  # Extract the text from the page
  content = r.text

# Use BeautifulSoup to read and parse the data as HTML
root = BeautifulSoup(content, 'html.parser')

# View the content and find the appropriate table
# print(root.prettify())
root.prettify()

# Use find() to save the aforementioned table as a variable
table = root.find('table')

# Use pandas to read in the HTML file.
df = pd.read_html(str(table))
# create data frame from list
df = pd.concat(df)

print(f"Dimension: {df.shape[0]} x {df.shape[1]} \n")
df.columns = ['rank', 'x_class', 'date', 'region', 'start_time', 'max_time', 'e
df.index = df.index + 1
df
```

```
Dimension: 50 x 8
```

Out[3]:

	rank	x_class	date	region	start_time	max_time	end_time	movie
1	1	X28+	2003/11/04	486	19:29	19:53	20:06	MovieView archive
2	2	X20+	2001/04/02	9393	21:32	21:51	22:03	MovieView archive
3	3	X17.2+	2003/10/28	486	09:51	11:10	11:24	MovieView archive
4	4	X17+	2005/09/07	808	17:17	17:40	18:03	MovieView archive
5	5	X14.4	2001/04/15	9415	13:19	13:50	13:55	MovieView archive
6	6	X10	2003/10/29	486	20:37	20:49	21:01	MovieView archive
7	7	X9.4	1997/11/06	8100	11:49	11:55	12:01	MovieView archive
8	8	X9.3	2017/09/06	2673	11:53	12:02	12:10	MovieView archive
9	9	X9	2006/12/05	930	10:18	10:35	10:45	MovieView archive
10	10	X8.3	2003/11/02	486	17:03	17:25	17:39	MovieView archive
11	11	X8.2	2017/09/10	2673	15:35	16:06	16:31	MovieView archive
12	12	X7.1	2005/01/20	720	06:36	07:01	07:26	MovieView archive
13	13	X6.9	2011/08/09	1263	07:48	08:05	08:08	MovieView archive
14	14	X6.5	2006/12/06	930	18:29	18:47	19:00	MovieView archive
15	15	X6.2	2005/09/09	808	19:13	20:04	20:36	MovieView archive
16	16	X6.2	2001/12/13	9733	14:20	14:30	14:35	MovieView archive
17	17	X5.7	2000/07/14	9077	10:03	10:24	10:43	MovieView archive
18	18	X5.6	2001/04/06	9415	19:10	19:21	19:31	MovieView archive
19	19	X5.4	2012/03/07	1429	00:02	00:24	00:40	MovieView archive
20	20	X5.4	2005/09/08	808	20:52	21:06	21:17	MovieView archive
21	21	X5.4	2003/10/23	486	08:19	08:35	08:49	MovieView archive
22	22	X5.3	2001/08/25	9591	16:23	16:45	17:04	MovieView archive
23	23	X4.9	2014/02/25	1990	00:39	00:49	01:03	MovieView archive
24	24	X4.9	1998/08/18	8307	22:10	22:19	22:28	View archive
25	25	X4.8	2002/07/23	39	00:18	00:35	00:47	MovieView archive
26	26	X4	2000/11/26	9236	16:34	16:48	16:56	MovieView archive
27	27	X3.9	2003/11/03	488	09:43	09:55	10:19	MovieView archive
28	28	X3.9	1998/08/19	8307	21:35	21:45	21:50	View archive
29	29	X3.8	2005/01/17	720	06:59	09:52	10:07	MovieView archive
30	30	X3.7	1998/11/22	8384	06:30	06:42	06:49	MovieView archive
31	31	X3.6	2005/09/09	808	09:42	09:59	10:08	MovieView archive
32	32	X3.6	2004/07/16	649	13:49	13:55	14:01	MovieView archive
33	33	X3.6	2003/05/28	365	00:17	00:27	00:39	MovieView archive
34	34	X3.4	2006/12/13	930	02:14	02:40	02:57	MovieView archive
35	35	X3.4	2001/12/28	9767	20:02	20:45	21:32	MovieView archive

	rank	x_class	date	region	start_time	max_time	end_time	movie
36	36	X3.3	2013/11/05	1890	22:07	22:12	22:15	MovieView archive
37	37	X3.3	2002/07/20	39	21:04	21:30	21:54	MovieView archive
38	38	X3.3	1998/11/28	8395	04:54	05:52	06:13	MovieView archive
39	39	X3.2	2013/05/14	1748	00:00	01:11	01:20	MovieView archive
40	40	X3.1	2014/10/24	2192	21:07	21:41	22:13	MovieView archive
41	41	X3.1	2002/08/24	69	00:49	01:12	01:31	MovieView archive
42	42	X3	2002/07/15	30	19:59	20:08	20:14	MovieView archive
43	43	X2.8	2013/05/13	1748	15:48	16:05	16:16	MovieView archive
44	44	X2.8	2001/12/11	9733	07:58	08:08	08:14	MovieView archive
45	45	X2.8	1998/08/18	8307	08:14	08:24	08:32	View archive
46	46	X2.7	2015/05/05	2339	22:05	22:11	22:15	MovieView archive
47	47	X2.7	2003/11/03	488	01:09	01:30	01:45	MovieView archive
48	48	X2.7	1998/05/06	8210	07:58	08:09	08:20	MovieView archive
49	49	X2.6	2005/01/15	720	22:25	23:02	23:31	MovieView archive
50	50	X2.6	2001/09/24	9632	09:32	10:38	11:09	MovieView archive

Step 2: Tidy the top 50 solar flare data

In [4]:

```
# Drop the last column of the table
df = df.drop(df.columns[-1], axis = 1).copy()

# Use datetime import to combine the date and each of the three time columns ir
for index, row in df.iterrows():
  start_datetime = row['date'] + ' ' + row['start_time']
  max_datetime = row['date'] + ' ' + row['max_time']
  end_datetime = row['date'] + ' ' + row['end_time']

  # Update the values in the dataframe
  df.at[index, 'start_datetime'] = pd.to_datetime(start_datetime)
  df.at[index, 'max_datetime'] = pd.to_datetime(max_datetime)
  df.at[index, 'end_datetime'] = pd.to_datetime(end_datetime)

# Drop the extra columns.
df = df.drop(['date', 'start_time', 'max_time', 'end_time'], axis = 1)

# Set regions coded as - as missing (NaN).
# After verifying the data, no missing values were found. However, the followir
df["region"] = df["region"].replace('-', np.nan)

# Change the order of columns
df = df[['rank','x_class','start_datetime','max_datetime','end_datetime','regic

print(f"A Dimension: {df.shape[0]} x {df.shape[1]} \n")
space_weather_top = df.head(50).copy()
df
```

A Dimension: 50 x 6

Out[4]:

	rank	x_class	start_datetime	max_datetime	end_datetime	region
1	1	X28+	2003-11-04 19:29:00	2003-11-04 19:53:00	2003-11-04 20:06:00	486
2	2	X20+	2001-04-02 21:32:00	2001-04-02 21:51:00	2001-04-02 22:03:00	9393
3	3	X17.2+	2003-10-28 09:51:00	2003-10-28 11:10:00	2003-10-28 11:24:00	486
4	4	X17+	2005-09-07 17:17:00	2005-09-07 17:40:00	2005-09-07 18:03:00	808
5	5	X14.4	2001-04-15 13:19:00	2001-04-15 13:50:00	2001-04-15 13:55:00	9415
6	6	X10	2003-10-29 20:37:00	2003-10-29 20:49:00	2003-10-29 21:01:00	486
7	7	X9.4	1997-11-06 11:49:00	1997-11-06 11:55:00	1997-11-06 12:01:00	8100
8	8	X9.3	2017-09-06 11:53:00	2017-09-06 12:02:00	2017-09-06 12:10:00	2673
9	9	X9	2006-12-05 10:18:00	2006-12-05 10:35:00	2006-12-05 10:45:00	930
10	10	X8.3	2003-11-02 17:03:00	2003-11-02 17:25:00	2003-11-02 17:39:00	486
11	11	X8.2	2017-09-10 15:35:00	2017-09-10 16:06:00	2017-09-10 16:31:00	2673
12	12	X7.1	2005-01-20 06:36:00	2005-01-20 07:01:00	2005-01-20 07:26:00	720
13	13	X6.9	2011-08-09 07:48:00	2011-08-09 08:05:00	2011-08-09 08:08:00	1263
14	14	X6.5	2006-12-06 18:29:00	2006-12-06 18:47:00	2006-12-06 19:00:00	930
15	15	X6.2	2005-09-09 19:13:00	2005-09-09 20:04:00	2005-09-09 20:36:00	808
16	16	X6.2	2001-12-13 14:20:00	2001-12-13 14:30:00	2001-12-13 14:35:00	9733
17	17	X5.7	2000-07-14 10:03:00	2000-07-14 10:24:00	2000-07-14 10:43:00	9077
18	18	X5.6	2001-04-06 19:10:00	2001-04-06 19:21:00	2001-04-06 19:31:00	9415
19	19	X5.4	2012-03-07 00:02:00	2012-03-07 00:24:00	2012-03-07 00:40:00	1429
20	20	X5.4	2005-09-08 20:52:00	2005-09-08 21:06:00	2005-09-08 21:17:00	808
21	21	X5.4	2003-10-23 08:19:00	2003-10-23 08:35:00	2003-10-23 08:49:00	486
22	22	X5.3	2001-08-25 16:23:00	2001-08-25 16:45:00	2001-08-25 17:04:00	9591
23	23	X4.9	2014-02-25 00:39:00	2014-02-25 00:49:00	2014-02-25 01:03:00	1990
24	24	X4.9	1998-08-18 22:10:00	1998-08-18 22:19:00	1998-08-18 22:28:00	8307
25	25	X4.8	2002-07-23 00:18:00	2002-07-23 00:35:00	2002-07-23 00:47:00	39
26	26	X4	2000-11-26 16:34:00	2000-11-26 16:48:00	2000-11-26 16:56:00	9236
27	27	X3.9	2003-11-03 09:43:00	2003-11-03 09:55:00	2003-11-03 10:19:00	488
28	28	X3.9	1998-08-19 21:35:00	1998-08-19 21:45:00	1998-08-19 21:50:00	8307
29	29	X3.8	2005-01-17 06:59:00	2005-01-17 09:52:00	2005-01-17 10:07:00	720
30	30	X3.7	1998-11-22 06:30:00	1998-11-22 06:42:00	1998-11-22 06:49:00	8384
31	31	X3.6	2005-09-09 09:42:00	2005-09-09 09:59:00	2005-09-09 10:08:00	808
32	32	X3.6	2004-07-16 13:49:00	2004-07-16 13:55:00	2004-07-16 14:01:00	649
33	33	X3.6	2003-05-28 00:17:00	2003-05-28 00:27:00	2003-05-28 00:39:00	365
34	34	X3.4	2006-12-13 02:14:00	2006-12-13 02:40:00	2006-12-13 02:57:00	930
35	35	X3.4	2001-12-28 20:02:00	2001-12-28 20:45:00	2001-12-28 21:32:00	9767

	rank	x_class	start_datetime	max_datetime	end_datetime	region
36	36	X3.3	2013-11-05 22:07:00	2013-11-05 22:12:00	2013-11-05 22:15:00	1890
37	37	X3.3	2002-07-20 21:04:00	2002-07-20 21:30:00	2002-07-20 21:54:00	39
38	38	X3.3	1998-11-28 04:54:00	1998-11-28 05:52:00	1998-11-28 06:13:00	8395
39	39	X3.2	2013-05-14 00:00:00	2013-05-14 01:11:00	2013-05-14 01:20:00	1748
40	40	X3.1	2014-10-24 21:07:00	2014-10-24 21:41:00	2014-10-24 22:13:00	2192
41	41	X3.1	2002-08-24 00:49:00	2002-08-24 01:12:00	2002-08-24 01:31:00	69
42	42	X3	2002-07-15 19:59:00	2002-07-15 20:08:00	2002-07-15 20:14:00	30
43	43	X2.8	2013-05-13 15:48:00	2013-05-13 16:05:00	2013-05-13 16:16:00	1748
44	44	X2.8	2001-12-11 07:58:00	2001-12-11 08:08:00	2001-12-11 08:14:00	9733
45	45	X2.8	1998-08-18 08:14:00	1998-08-18 08:24:00	1998-08-18 08:32:00	8307
46	46	X2.7	2015-05-05 22:05:00	2015-05-05 22:11:00	2015-05-05 22:15:00	2339
47	47	X2.7	2003-11-03 01:09:00	2003-11-03 01:30:00	2003-11-03 01:45:00	488
48	48	X2.7	1998-05-06 07:58:00	1998-05-06 08:09:00	1998-05-06 08:20:00	8210
49	49	X2.6	2005-01-15 22:25:00	2005-01-15 23:02:00	2005-01-15 23:31:00	720
50	50	X2.6	2001-09-24 09:32:00	2001-09-24 10:38:00	2001-09-24 11:09:00	9632

Step 3: Scrape the NASA data

In [5]:

```
# Read from NASA
r = requests.get("http://cdaw.gsfc.nasa.gov/CME_list/radio/waves_type2.html")

# Check status code
if r.status_code != 200:
  print("Request failed from the provided website.")
else:
  # Extract the text from the page
  content = r.text

# Use BeautifulSoup to read and parse the data as HTML
root = BeautifulSoup(content, 'html.parser')

# View the content and find the appropriate table
# print(root.prettify())
root.prettify()

# Data are in the HTML tag of <pre>, store it as the text
data = root.find('pre').text

# Remove all description sentences using slice, lines are splited by newline ch
# to obtain each row of data as a long string.
data = data.split('\n')[12:-2]

# create data frame from list
df = pd.DataFrame(data)

# As each row of data are stored in ID of 0 as a long string,
```

```
# we need to split that String column into multiple columns
df = df[0].str.split(expand = True)

# Drop comments at the end
cols = [15,16,17,18,19,20,21,22,23]
df = df.drop(df.columns[cols], axis = 1)

print(f"Dimension: {df.shape[0]} x {df.shape[1]} \n")
df.columns = ['start_date', 'start_time', 'end_date', 'end_time', 'start_freque
              'flare_location', 'flare_region', 'importance', 'cme_date', 'cme_
df.index = df.index + 1
df
```

```
Dimension: 522 x 15
```

Out[5]:

	start_date	start_time	end_date	end_time	start_frequency	end_frequency	flare_locatio
1	1997/04/01	14:00	04/01	14:15	8000	4000	S25E1
2	1997/04/07	14:30	04/07	17:30	11000	1000	S28E1
3	1997/05/12	05:15	05/14	16:00	12000	80	N21W0
4	1997/05/21	20:20	05/21	22:00	5000	500	N05W1
5	1997/09/23	21:53	09/23	22:16	6000	2000	S29E2
...	...	...	...	...	...	...	
518	2017/09/17	11:45	09/17	12:35	16000	900	S08E17
519	2017/10/18	05:48	10/18	12:40	16000	400	S06E12
520	2019/05/03	23:52	05/04	00:16	13000	2300	N12E8
521	2020/11/29	13:07	11/29	15:23	14000	850	S23E8
522	2020/12/07	16:18	12/08	02:00	14000	160	S25W0

522 rows × 15 columns

Step 4: Tidy the NASA table

In [6]:

```
import re

df = df.copy()
# Recode any missing entries as NaN
df.replace(['????','-----','---','------','BACK','Back','Back?', \
            'FILA','DSF','altr','----','--/--','--:--'], np.nan, inplace = True

# Create a new column that indicates if a row corresponds to a halo flare or no
# and then replace Halo entries in the cme_angle column as NA.
df['is_halo'] = np.where(df['cpa']=='Halo', True, False)
df['cpa'] = df['cpa'].replace('Halo', 'NA')

# Create a new column that indicates if width is given as a lower bound,
# and remove any non-numeric part of the width column.
df['width_lower_bound'] = df['width'].str.startswith('>',na=False)
df['width'] = df['width'].str.replace('>','')

# Combine date and time columns for start, end and cme so they can be encoded a
```

```
for index, row in df.iterrows():

  start_datetime = row['start_date'] + ' ' + row['start_time']
  year = start_datetime.split('/')[0]
  # Update the values in the dataframe
  df.at[index, 'start_datetime'] = pd.to_datetime(start_datetime)

  # Pay attention to end_datetime as time format are different
  if row['end_time'] == '24:00':
    # Fix time format unmatch error 24:00 -> 00:00
    row['end_time'] = row['end_time'].replace('24:00', '00:00')
    end_datetime = year + '/' + row['end_date'] + ' ' + row['end_time']
    # Update the values in the dataframe
    # after check all datas, we can directly add 1 to dates, there are no speci
    df.at[index, 'end_datetime'] = pd.to_datetime(end_datetime) + pd.to_timedel
  else:
    end_datetime = year + '/' + row['end_date'] + ' ' + row['end_time']
    # Update the values in the dataframe
    df.at[index, 'end_datetime'] = pd.to_datetime(end_datetime)

  # Pay attention to cme_datetime as they might be empty
  if row['cme_date'] is not np.nan and row['cme_time'] is not np.nan:
    cme_datetime = year + '/' + row['cme_date'] + ' ' + row['cme_time']
    # Update the values in the dataframe
    df.at[index, 'cme_datetime'] = pd.to_datetime(cme_datetime)

# Drop the extra columns.
df = df.drop(['start_date', 'start_time', 'end_date', 'end_time', 'cme_date', '

# Change the order of columns
df = df[['start_datetime', 'end_datetime', 'start_frequency', 'end_frequency',
         'flare_region', 'importance', 'cme_datetime', 'cpa', 'width', 'speed'

nasa = df.copy()
df
```

Out[6]:

	start_datetime	end_datetime	start_frequency	end_frequency	flare_location	flare_region
1	1997-04-01 14:00:00	1997-04-01 14:15:00	8000	4000	S25E16	8026
2	1997-04-07 14:30:00	1997-04-07 17:30:00	11000	1000	S28E19	8027
3	1997-05-12 05:15:00	1997-05-14 16:00:00	12000	80	N21W08	8038
4	1997-05-21 20:20:00	1997-05-21 22:00:00	5000	500	N05W12	8040
5	1997-09-23 21:53:00	1997-09-23 22:16:00	6000	2000	S29E25	8088
...	...	...	...	...	...	...
518	2017-09-17 11:45:00	2017-09-17 12:35:00	16000	900	S08E170	NaN
519	2017-10-18 05:48:00	2017-10-18 12:40:00	16000	400	S06E123	NaN
520	2019-05-03 23:52:00	2019-05-04 00:16:00	13000	2300	N12E82	12740
521	2020-11-29 13:07:00	2020-11-29 15:23:00	14000	850	S23E89	NaN
522	2020-12-07 16:18:00	2020-12-08 02:00:00	14000	160	S25W08	12790

522 rows × 14 columns

Part 2: Analysis

Question 1: Replication (10 pts)

In [7]:

```
# There are more than 50 rows start with letter X for importance,
# So, we can grab that data and sort it to get top 50 solar flares
data_X = df[df['importance'].str.startswith('X', na=False)].copy()

# Remove X, and then sort the data based on the float numbers
# We create a new column to do this in order to keep the initial data
data_X['remove_X'] = data_X['importance'].str.replace('X', '')
data_X['remove_X'] = data_X['remove_X'].astype(float)

# Take top 50 data
top_50 = data_X.sort_values(by="remove_X",ascending=False).head(50)
top_50 = top_50.drop(top_50.columns[-1], axis = 1)
nasa_top = top_50.copy()
top_50
```

Out[7]:

	start_datetime	end_datetime	start_frequency	end_frequency	flare_location	flare_regior
241	2003-11-04 20:00:00	2003-11-05 00:00:00	10000	200	S19W83	10486
118	2001-04-02 22:05:00	2001-04-03 02:30:00	14000	250	N19W72	9393
234	2003-10-28 11:10:00	2003-10-30 00:00:00	14000	40	S16E08	10486
127	2001-04-15 14:05:00	2001-04-16 13:00:00	14000	40	S20W85	9415
235	2003-10-29 20:55:00	2003-10-30 00:00:00	11000	500	S15W02	10486
9	1997-11-06 12:20:00	1997-11-07 08:30:00	14000	100	S18W63	8100
515	2017-09-06 12:05:00	2017-09-07 08:00:00	16000	70	S08W33	12673
329	2006-12-05 10:50:00	2006-12-05 20:00:00	14000	250	S07E68	10930
238	2003-11-02 17:30:00	2003-11-03 01:00:00	12000	250	S14W56	10486
516	2017-09-10 16:02:00	2017-09-11 06:50:00	16000	150	S09W92	NaN
289	2005-01-20 07:15:00	2005-01-20 16:30:00	14000	25	N14W61	10720
360	2011-08-09 08:20:00	2011-08-09 08:35:00	16000	4000	N17W69	11263
332	2006-12-06 19:00:00	2006-12-09 00:00:00	16000	30	S05E64	10930
318	2005-09-09 19:45:00	2005-09-09 22:00:00	10000	50	S12E67	10808
83	2000-07-14 10:30:00	2000-07-15 14:30:00	14000	80	N22W07	9077
122	2001-04-06 19:35:00	2001-04-07 01:50:00	14000	230	S21E31	9415
376	2012-03-07 01:00:00	2012-03-08 19:00:00	16000	30	N17E27	11429
136	2001-08-25 16:50:00	2001-08-25 23:00:00	8000	170	S17E34	9591
444	2014-02-25 00:56:00	2014-02-25 11:28:00	14000	100	S12E82	11990
194	2002-07-23 00:50:00	2002-07-23 04:00:00	11000	400	S13E72	10039
105	2000-11-26 17:00:00	2000-11-26 17:15:00	14000	7000	N18W38	9236
240	2003-11-03 10:00:00	2003-11-03 12:30:00	6000	400	N08W77	10488

	start_datetime	end_datetime	start_frequency	end_frequency	flare_location	flare_regior
287	2005-01-17 10:00:00	2005-01-17 10:35:00	6100	1500	N15W25	10720
223	2003-05-28 01:00:00	2003-05-29 00:30:00	1000	200	S07W20	10365
333	2006-12-13 02:45:00	2006-12-13 10:40:00	12000	150	S06W23	10930
161	2001-12-28 20:35:00	2001-12-29 03:00:00	14000	350	S26E90	9756
193	2002-07-20 21:30:00	2002-07-20 22:20:00	10000	2000	S13E90	10039
405	2013-05-14 01:16:00	2013-05-14 08:20:00	16000	240	N08E77	11748
202	2002-08-24 01:45:00	2002-08-24 03:25:00	5000	400	S02W81	10069
404	2013-05-13 16:15:00	2013-05-13 19:10:00	16000	300	N11E85	11748
488	2015-05-05 22:24:00	2015-05-05 23:14:00	14000	500	N15E79	12339
20	1998-05-06 08:25:00	1998-05-06 08:35:00	14000	5000	S11W65	8210
239	2003-11-03 01:15:00	2003-11-03 01:25:00	3000	1500	N10W83	10488
285	2005-01-15 23:00:00	2005-01-17 00:00:00	3000	40	N15W05	10720
143	2001-09-24 10:45:00	2001-09-25 20:00:00	7000	30	S16E23	9632
10	1997-11-27 13:30:00	1997-11-27 14:00:00	14000	7000	N17E63	8113
277	2004-11-10 02:25:00	2004-11-10 03:40:00	14000	1000	N09W49	10696
124	2001-04-10 05:24:00	2001-04-11 00:00:00	14000	100	S23W09	9415
100	2000-11-24 15:25:00	2000-11-24 22:00:00	14000	200	N22W07	9236
74	2000-06-06 15:20:00	2000-06-08 09:00:00	14000	40	N20E18	9026
346	2011-02-15 02:10:00	2011-02-15 07:00:00	16000	400	S20W12	11158
319	2005-09-10 21:45:00	2005-09-11 01:00:00	14000	200	S13E47	10808
362	2011-09-06 22:30:00	2011-09-07 15:40:00	16000	150	N14W18	11283
421	2013-10-25 15:08:00	2013-10-25 22:32:00	16000	200	S06E69	11882

	start_datetime	end_datetime	start_frequency	end_frequency	flare_location	flare_regior
8	1997-11-04 06:00:00	1997-11-05 04:30:00	14000	100	S14W33	8100
99	2000-11-24 05:10:00	2000-11-24 15:00:00	14000	100	N20W05	9236
126	2001-04-12 10:20:00	2001-04-12 10:40:00	14000	7000	S19W43	9415
275	2004-11-07 16:25:00	2004-11-08 20:00:00	14000	60	N09W17	10696
286	2005-01-17 09:25:00	2005-01-17 16:00:00	14000	30	N15W25	10720
103	2000-11-25 19:00:00	2000-11-25 19:35:00	6000	2000	N20W23	9236

After comparing the data we obtained from NASA and the data we visualized in part 1 from SpaceWeatherLive, it can be concluded that while most of the data can be matched, they are not exactly the same. SpaceWeatherLive had some data that NASA did not provide or observe, and there were also some small value differences due to potential differences in observation methods.

The code provided above sorts the data in the NASA table to find the top 50 solar flares. The classification of the flares consists of a letter followed by a floating number. To sort the data by this classification, we first group the flares by the letter component and select the group with the largest letter, which is 'X'. Then, we compare the floating number within this group to obtain the top 50 flares.

Question 2: Integration

In [8]:
```
# array to record the ranking of current row in NASA according to SpaceWeatherl
top_50['Ranking in SpaceWeatherLive'] = 0

# Determine match
for index_s, row_s in space_weather_top.iterrows():
  for index_n, row_n in nasa_top.iterrows():
    if abs((row_n['start_datetime']-row_s['start_datetime']).total_seconds()) <
        and abs((row_n['end_datetime']-row_s['end_datetime']).total_seconds())
      top_50.at[index_n, 'Ranking in SpaceWeatherLive'] = index_s

top_50
```

Out[8]:

	start_datetime	end_datetime	start_frequency	end_frequency	flare_location	flare_regior
241	2003-11-04 20:00:00	2003-11-05 00:00:00	10000	200	S19W83	10486
118	2001-04-02 22:05:00	2001-04-03 02:30:00	14000	250	N19W72	9393
234	2003-10-28 11:10:00	2003-10-30 00:00:00	14000	40	S16E08	10486
127	2001-04-15 14:05:00	2001-04-16 13:00:00	14000	40	S20W85	9415
235	2003-10-29 20:55:00	2003-10-30 00:00:00	11000	500	S15W02	10486
9	1997-11-06 12:20:00	1997-11-07 08:30:00	14000	100	S18W63	8100
515	2017-09-06 12:05:00	2017-09-07 08:00:00	16000	70	S08W33	12673
329	2006-12-05 10:50:00	2006-12-05 20:00:00	14000	250	S07E68	10930
238	2003-11-02 17:30:00	2003-11-03 01:00:00	12000	250	S14W56	10486
516	2017-09-10 16:02:00	2017-09-11 06:50:00	16000	150	S09W92	NaN
289	2005-01-20 07:15:00	2005-01-20 16:30:00	14000	25	N14W61	10720
360	2011-08-09 08:20:00	2011-08-09 08:35:00	16000	4000	N17W69	11263
332	2006-12-06 19:00:00	2006-12-09 00:00:00	16000	30	S05E64	10930
318	2005-09-09 19:45:00	2005-09-09 22:00:00	10000	50	S12E67	10808
83	2000-07-14 10:30:00	2000-07-15 14:30:00	14000	80	N22W07	9077
122	2001-04-06 19:35:00	2001-04-07 01:50:00	14000	230	S21E31	9415
376	2012-03-07 01:00:00	2012-03-08 19:00:00	16000	30	N17E27	11429
136	2001-08-25 16:50:00	2001-08-25 23:00:00	8000	170	S17E34	9591
444	2014-02-25 00:56:00	2014-02-25 11:28:00	14000	100	S12E82	11990
194	2002-07-23 00:50:00	2002-07-23 04:00:00	11000	400	S13E72	10039
105	2000-11-26 17:00:00	2000-11-26 17:15:00	14000	7000	N18W38	9236
240	2003-11-03 10:00:00	2003-11-03 12:30:00	6000	400	N08W77	10488

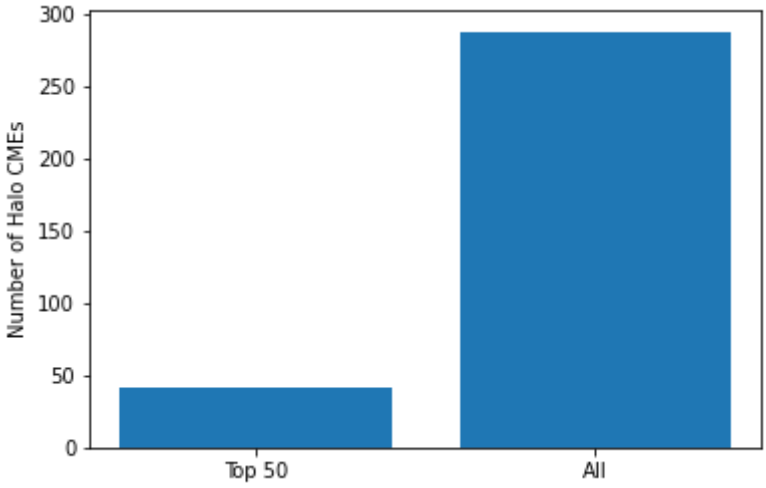
	start_datetime	end_datetime	start_frequency	end_frequency	flare_location	flare_regior
287	2005-01-17 10:00:00	2005-01-17 10:35:00	6100	1500	N15W25	10720
223	2003-05-28 01:00:00	2003-05-29 00:30:00	1000	200	S07W20	10365
333	2006-12-13 02:45:00	2006-12-13 10:40:00	12000	150	S06W23	10930
161	2001-12-28 20:35:00	2001-12-29 03:00:00	14000	350	S26E90	9756
193	2002-07-20 21:30:00	2002-07-20 22:20:00	10000	2000	S13E90	10039
405	2013-05-14 01:16:00	2013-05-14 08:20:00	16000	240	N08E77	11748
202	2002-08-24 01:45:00	2002-08-24 03:25:00	5000	400	S02W81	10069
404	2013-05-13 16:15:00	2013-05-13 19:10:00	16000	300	N11E85	11748
488	2015-05-05 22:24:00	2015-05-05 23:14:00	14000	500	N15E79	12339
20	1998-05-06 08:25:00	1998-05-06 08:35:00	14000	5000	S11W65	8210
239	2003-11-03 01:15:00	2003-11-03 01:25:00	3000	1500	N10W83	10488
285	2005-01-15 23:00:00	2005-01-17 00:00:00	3000	40	N15W05	10720
143	2001-09-24 10:45:00	2001-09-25 20:00:00	7000	30	S16E23	9632
10	1997-11-27 13:30:00	1997-11-27 14:00:00	14000	7000	N17E63	8113
277	2004-11-10 02:25:00	2004-11-10 03:40:00	14000	1000	N09W49	10696
124	2001-04-10 05:24:00	2001-04-11 00:00:00	14000	100	S23W09	9415
100	2000-11-24 15:25:00	2000-11-24 22:00:00	14000	200	N22W07	9236
74	2000-06-06 15:20:00	2000-06-08 09:00:00	14000	40	N20E18	9026
346	2011-02-15 02:10:00	2011-02-15 07:00:00	16000	400	S20W12	11158
319	2005-09-10 21:45:00	2005-09-11 01:00:00	14000	200	S13E47	10808
362	2011-09-06 22:30:00	2011-09-07 15:40:00	16000	150	N14W18	11283
421	2013-10-25 15:08:00	2013-10-25 22:32:00	16000	200	S06E69	11882

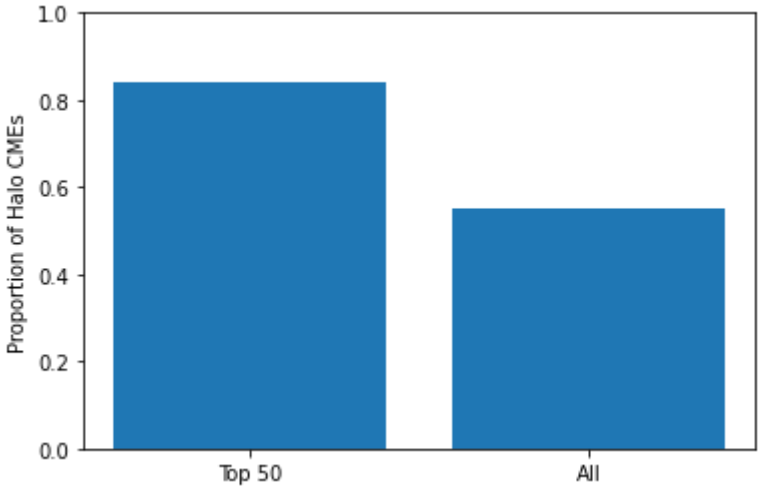
	start_datetime	end_datetime	start_frequency	end_frequency	flare_location	flare_regior
8	1997-11-04 06:00:00	1997-11-05 04:30:00	14000	100	S14W33	8100
99	2000-11-24 05:10:00	2000-11-24 15:00:00	14000	100	N20W05	9236
126	2001-04-12 10:20:00	2001-04-12 10:40:00	14000	7000	S19W43	9415
275	2004-11-07 16:25:00	2004-11-08 20:00:00	14000	60	N09W17	10696
286	2005-01-17 09:25:00	2005-01-17 16:00:00	14000	30	N15W25	10720
103	2000-11-25 19:00:00	2000-11-25 19:35:00	6000	2000	N20W23	9236

I defined the best matched row are start time and end time are in the time range of 1 day. If the data from NASA and the data from the SpaceWeatherLive were record a solar flare event that the difference of start time and end time are both not greater than 1 day, then they can be considered as same event.

Question 3: Analysis

In [9]:
```
import matplotlib.pyplot as plt

# proportion of Halo CMEs in the top 50 flares
halo_top_50 = top_50[top_50['is_halo']]
halo_top_50_count = len(halo_top_50)
halo_top_50_prop = halo_top_50_count / len(top_50)

# proportion of Halo CMEs in all flares
halo_all = nasa[nasa['is_halo']]
halo_all_count = len(halo_all)
halo_all_prop = halo_all_count / len(df)

labels = ['Top 50', 'All']
counts = [halo_top_50_count, halo_all_count]
proportions = [halo_top_50_prop, halo_all_prop]

fig, ax = plt.subplots()
ax.bar(labels, counts)
ax.set_ylabel('Number of Halo CMEs')

fig, ax = plt.subplots()
ax.bar(labels, proportions)
ax.set_ylabel('Proportion of Halo CMEs')
ax.set_ylim(0, 1)

plt.show()
```





(a). The intent of the plot is to compare the number or proportion of Halo CMEs in the top 50 flares with the entire dataset, in order to investigate whether flares in the top 50 tend to have Halo CMEs by comparing the proportion. (b). Code shows above. (c). The first plot shows total number of Halo CMEs both in Top50 and all data. The second plot shows the proportion of Halo CMEs both in Top50 and all data. We can eaily see that Top50 have higher proportion though the number of it is small. (d). The plot shows that the proportion of Halo CMEs in the top 50 flares is lower than the proportion in the entire dataset, suggesting that flares in the top 50 do not tend to have Halo CMEs.