**Space X Falcon 9 First Stage Landing Prediction**

**Web scraping Falcon 9 and Falcon Heavy Launches Records from Wikipedia**

Estimated time needed: **40** minutes

In this lab, you will be performing web scraping to collect Falcon 9 historical launch records from a Wikipedia page titled List of Falcon 9 and Falcon Heavy launches

[https://en.wikipedia.org/wiki/List\_of\_Falcon\\_9\\_and\_Falcon\_Heavy\_launches](https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches?utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_content=000026UJ&utm_term=10006555&utm_id=NA-SkillsNetwork-Channel-SkillsNetworkCoursesIBMDS0321ENSkillsNetwork26802033-2022-01-01)

Falcon 9 first stage will land successfully



Several examples of an unsuccessful landing are shown here:

More specifically, the launch records are stored in a HTML table shown below:



**Objectives**

Web scrap Falcon 9 launch records with BeautifulSoup:

* Extract a Falcon 9 launch records HTML table from Wikipedia
* Parse the table and convert it into a Pandas data frame

First let's import required packages for this lab

In [1]:

**!**pip3 install beautifulsoup4

**!**pip3 install requests

**!**pip3 install html5lib

Requirement already satisfied: beautifulsoup4 in /opt/conda/envs/Python-3.10/lib/python3.10/site-packages (4.11.1)

Requirement already satisfied: soupsieve>1.2 in /opt/conda/envs/Python-3.10/lib/python3.10/site-packages (from beautifulsoup4) (2.3.1)

Requirement already satisfied: requests in /opt/conda/envs/Python-3.10/lib/python3.10/site-packages (2.28.1)

Requirement already satisfied: certifi>=2017.4.17 in /opt/conda/envs/Python-3.10/lib/python3.10/site-packages (from requests) (2022.9.24)

Requirement already satisfied: charset-normalizer<3,>=2 in /opt/conda/envs/Python-3.10/lib/python3.10/site-packages (from requests) (2.0.4)

Requirement already satisfied: urllib3<1.27,>=1.21.1 in /opt/conda/envs/Python-3.10/lib/python3.10/site-packages (from requests) (1.26.11)

Requirement already satisfied: idna<4,>=2.5 in /opt/conda/envs/Python-3.10/lib/python3.10/site-packages (from requests) (3.3)

Collecting html5lib

Downloading html5lib-1.1-py2.py3-none-any.whl (112 kB)

━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 112.2/112.2 kB 10.7 MB/s eta 0:00:00

Collecting webencodings

Downloading webencodings-0.5.1-py2.py3-none-any.whl (11 kB)

Requirement already satisfied: six>=1.9 in /opt/conda/envs/Python-3.10/lib/python3.10/site-packages (from html5lib) (1.16.0)

Installing collected packages: webencodings, html5lib

Successfully installed html5lib-1.1 webencodings-0.5.1

In [2]:

**import** sys

**import** requests

**from** bs4 **import** BeautifulSoup

**import** re

**import** unicodedata

**import** pandas **as** pd

and we will provide some helper functions for you to process web scraped HTML table

In [3]:

**def** date\_time(table\_cells):

"""

This function returns the data and time from the HTML table cell

Input: the element of a table data cell extracts extra row

"""

**return** [data\_time**.**strip() **for** data\_time **in** list(table\_cells**.**strings)][0:2]

**def** booster\_version(table\_cells):

"""

This function returns the booster version from the HTML table cell

Input: the element of a table data cell extracts extra row

"""

out**=**''**.**join([booster\_version **for** i,booster\_version **in** enumerate( table\_cells**.**strings) **if** i**%2**==0][0:-1])

**return** out

**def** landing\_status(table\_cells):

"""

This function returns the landing status from the HTML table cell

Input: the element of a table data cell extracts extra row

"""

out**=**[i **for** i **in** table\_cells**.**strings][0]

**return** out

**def** get\_mass(table\_cells):

mass**=**unicodedata**.**normalize("NFKD", table\_cells**.**text)**.**strip()

**if** mass:

mass**.**find("kg")

new\_mass**=**mass[0:mass**.**find("kg")**+**2]

**else**:

new\_mass**=**0

**return** new\_mass

**def** extract\_column\_from\_header(row):

"""

This function returns the landing status from the HTML table cell

Input: the element of a table data cell extracts extra row

"""

**if** (row**.**br):

row**.**br**.**extract()

**if** row**.**a:

row**.**a**.**extract()

**if** row**.**sup:

row**.**sup**.**extract()

colunm\_name **=** ' '**.**join(row**.**contents)

*# Filter the digit and empty names*

**if** **not**(colunm\_name**.**strip()**.**isdigit()):

colunm\_name **=** colunm\_name**.**strip()

**return** colunm\_name