SpaceX Falcon 9 first stage Landing Prediction

Collecting the data:

Request to the SpaceX API

Clean the requested data

```
import requests
import pandas as pd
import datetime
pd.set_option('display.max_columns', None)
pd.set_option('display.max_colwidth', None)

def getBoosterVersion(data):
    for x in data['rocket']:
       if x:
        response = requests.get("https://api.spacexdata.com/v4/rockets/"+str(x)).json()
        BoosterVersion.append(response['name'])

def getLaunchSite(data):
    for x in data['launchpad']:
       if x:
         response = requests.get("https://api.spacexdata.com/v4/launchpads/"+str(x)).json()
         Longitude.append(response['longitude'])
         Latitude.append(response['latitude'])
         LaunchSite.append(response['name'])

def getPayloadData(data):
    for load in data['payloads']:
       if load:
        response = requests.get("https://api.spacexdata.com/v4/payloads/"+load).json()
        PayloadMass.append(response['mass_kg'])
        Orbit.append(response['orbit'])

def getCoreData(data):
    for core in data['cores']:
            if core['core'] != None:
                response = requests.get("https://api.spacexdata.com/v4/cores/"+core['core']).json()
                Block.append(response['block'])
                ReusedCount.append(response['reuse_count'])
                Serial.append(response['serial'])
            else:
                Block.append(None)
                ReusedCount.append(None)
                Serial.append(None)
            Outcome.append(str(core['landing_success'])+'
'+str(core['landing_type']))
            Flights.append(core['flight'])
            GridFins.append(core['gridfins'])
            Reused.append(core['reused'])
            Legs.append(core['legs'])
            LandingPad.append(core['landpad'])

spacex_url="https://api.spacexdata.com/v4/launches/past"

response = requests.get(spacex_url)

# print(response.content)

static_json_url='https://cf-courses-data.s3.us.cloud-object-
storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/
API_call_spacex_api.json'

response.status_code
```

```
200
```

```
data = pd.json_normalize(response.json())  # convert to flat table
# print(data.head())

# Lets take a subset of our dataframe keeping only the features we
want and the flight number, and date_utc.
data = data[['rocket', 'payloads', 'launchpad', 'cores',
'flight_number', 'date_utc']]

# We will remove rows with multiple cores because those are falcon
rockets with 2 extra rocket boosters and rows that have multiple
payloads in a single rocket.
data = data[data['cores'].map(len)==1]
data = data[data['payloads'].map(len)==1]

# Since payloads and cores are lists of size 1 we will also extract
the single value in the list and replace the feature.
data['cores'] = data['cores'].map(lambda x : x[0])
data['payloads'] = data['payloads'].map(lambda x : x[0])

# We also want to convert the date_utc to a datetime datatype and then
extracting the date leaving the time
data['date'] = pd.to_datetime(data['date_utc']).dt.date

# Using the date we will restrict the dates of the launches
data = data[data['date'] <= datetime.date(2020, 11, 13)]
```

```python
# Global variables
BoosterVersion = []
PayloadMass = []
Orbit = []
LaunchSite = []
Outcome = []
Flights = []
GridFins = []
Reused = []
Legs = []
LandingPad = []
Block = []
ReusedCount = []
Serial = []
Longitude = []
Latitude = []

# Call getBoosterVersion
getBoosterVersion(data)
BoosterVersion[0:5]
```

```
['Falcon 1', 'Falcon 1', 'Falcon 1', 'Falcon 1', 'Falcon 9']
```

```python
# Call getLaunchSite
getLaunchSite(data)

# Call getPayloadData
getPayloadData(data)

# Call getCoreData
getCoreData(data)

# combine the columns into a dictionary
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
```

```
data_falcon9 = pd.DataFrame(launch_dict)
# print(data_falcon9.head())

print(data_falcon9['BoosterVersion'].value_counts())
data_falcon9 = data_falcon9[data_falcon9['BoosterVersion']=='Falcon
9']
print(data_falcon9['BoosterVersion'].value_counts())
```

```
Falcon 9    90
Falcon 1     4
Name: BoosterVersion, dtype: int64
Falcon 9    90
Name: BoosterVersion, dtype: int64
```

```
data_falcon9.loc[:,'FlightNumber'] = list(range(1,
data_falcon9.shape[0]+1))
data_falcon9
```

```
    FlightNumber        Date BoosterVersion  PayloadMass Orbit
LaunchSite  \
4              1  2010-06-04       Falcon 9          NaN   LEO  CCSFS
SLC 40
5              2  2012-05-22       Falcon 9        525.0   LEO  CCSFS
SLC 40
6              3  2013-03-01       Falcon 9        677.0   ISS  CCSFS
SLC 40
7              4  2013-09-29       Falcon 9        500.0    PO   VAFB
SLC 4E
8              5  2013-12-03       Falcon 9       3170.0   GTO  CCSFS
SLC 40
..           ...         ...            ...          ...   ...
...
89            86  2020-09-03       Falcon 9      15600.0  VLEO    KSC
LC 39A
90            87  2020-10-06       Falcon 9      15600.0  VLEO    KSC
LC 39A
91            88  2020-10-18       Falcon 9      15600.0  VLEO    KSC
LC 39A
92            89  2020-10-24       Falcon 9      15600.0  VLEO  CCSFS
SLC 40
93            90  2020-11-05       Falcon 9       3681.0   MEO  CCSFS
SLC 40

        Outcome  Flights  GridFins  Reused   Legs
LandingPad  \
4     None None        1     False   False  False
None
5     None None        1     False   False  False
None
6     None None        1     False   False  False
```

```
None
7   False Ocean        1     False   False  False
None
8     None None        1     False   False  False
None
..          ...      ...       ...     ...    ...
...
89    True ASDS        2      True    True   True
5e9e3032383ecb6bb234e7ca
90    True ASDS        3      True    True   True
5e9e3032383ecb6bb234e7ca
91    True ASDS        6      True    True   True
5e9e3032383ecb6bb234e7ca
92    True ASDS        3      True    True   True
5e9e3033383ecbb9e534e7cc
93    True ASDS        1      True   False   True
5e9e3032383ecb6bb234e7ca

    Block  ReusedCount Serial   Longitude   Latitude
4     1.0            0  B0003  -80.577366  28.561857
5     1.0            0  B0005  -80.577366  28.561857
6     1.0            0  B0007  -80.577366  28.561857
7     1.0            0  B1003 -120.610829  34.632093
8     1.0            0  B1004  -80.577366  28.561857
..    ...          ...    ...         ...        ...
89    5.0           12  B1060  -80.603956  28.608058
90    5.0           13  B1058  -80.603956  28.608058
91    5.0           12  B1051  -80.603956  28.608058
92    5.0           12  B1060  -80.577366  28.561857
93    5.0            8  B1062  -80.577366  28.561857

[90 rows x 17 columns]
```