SpaceX Falcon 9 first stage Landing Prediction

Lab 1: Collecting the data

Estimated time needed: **45** minutes

In this capstone, we will predict if the Falcon 9 first stage will land successfully. SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage. Therefore if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch. In this lab, you will collect and make sure the data is in the correct format from an API. The following is an example of a successful and launch.

Several examples of an unsuccessful landing are shown here:

Most unsuccessful landings are planned. Space X performs a controlled landing in the oceans.

Objectives

In this lab, you will make a get request to the SpaceX API. You will also do some basic data wrangling and formating.

- Request to the SpaceX API
- Clean the requested data

Import Libraries and Define Auxiliary Functions

We will import the following libraries into the lab

In [1]:

```
# Requests allows us to make HTTP requests which we will use to get data
from an API

import requests

# Pandas is a software library written for the Python programming language
for data manipulation and analysis.

import pandas as pd

# NumPy is a library for the Python programming language, adding support
for large, multi-dimensional arrays and matrices, along with a large
collection of high-level mathematical functions to operate on these arrays

import numpy as np

# Datetime is a library that allows us to represent dates

import datetime

# Setting this option will print all collumns of a dataframe

pd.set_option('display.max_columns', None)

# Setting this option will print all of the data in a feature

pd.set_option('display.max_colwidth', None)
```

Below we will define a series of helper functions that will help us use the API to extract information using identification numbers in the launch data.

From the rocket column we would like to learn the booster name.

In [2]:

```
# Takes the dataset and uses the rocket column to call the API and append
the data to the list

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'])
```

From the launchpad we would like to know the name of the launch site being used, the logitude, and the latitude.

In [3]:

```
# Takes the dataset and uses the launchpad column to call the API and
append the data to the list

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'])
```

From the payload we would like to learn the mass of the payload and the orbit that it is going to.

In [4]:

```
# Takes the dataset and uses the payloads column to call the API and append
the data to the lists

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'])
```

From cores we would like to learn the outcome of the landing, the type of the landing, number of flights with that core, whether gridfins were used, wheter the core is reused, wheter legs were used, the landing pad used, the block of the core which is a number used to seperate version of cores, the number of times this specific core has been reused, and the serial of the core.

In [5]:

```
# Takes the dataset and uses the cores column to call the API and append
the data to the lists

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'])
```