IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Aryna Artsiukevich
18.12.2023

Outline

Executive Summary

- Summary of methodologies
  - Data Collection with an API
  - Data Collection with Web Scraping
  - Data Wrangling
  - EDA with SQL
  - EDA: Data Visualization
  - EDA: Interactive Data Visualization
  - ML: Predictive Analysis

Introduction

- SpaceX is a company, making space travel affordable for everyone. Its accomplishments include: sending spacecraft to the International Space Station, providing satellite Internet access, sending manned missions to Space. One reason SpaceX can do this is the rocket launches are relatively inexpensive ($62 million), because it can reuse the first stage. This stage does most of the work and is much larger than the second stage. Sometimes the first stage does not land, sometimes it will crash as shown in this clip.
- Therefore, if we can determine if the first stage will land, we can determine the cost of a launch. Instead of using rocket science to determine if the first stage will land successfully, the goal is to train a machine learning model and use public information to predict if SpaceX will reuse the first stage.

Section 1

Methodology

Methodology

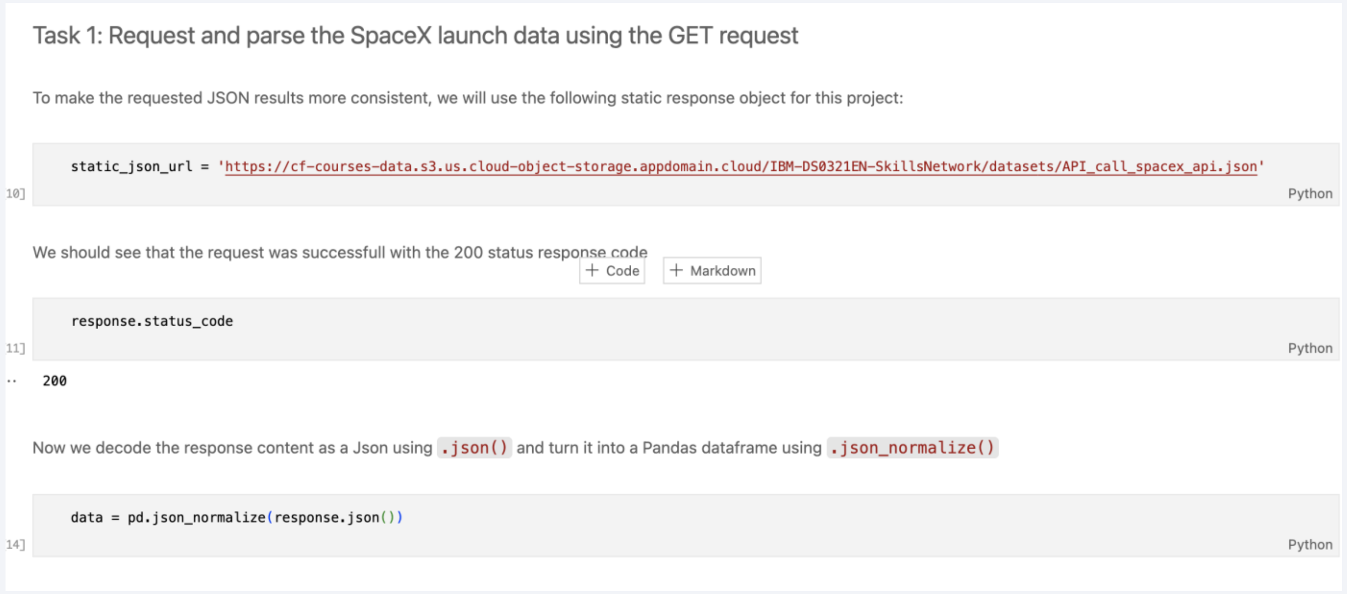
Executive Summary

- Data collection methodology:
    - Using SpaceX REST API and web-scrapping
- Perform data wrangling
    - Handling missing values, applying OHE for categorical features
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models

Data Collection – SpaceX API

- Collected data using Get request and cleaned the data

- GitHub link: https://github.com/ArinaArtiukevich/ibm_ds_professional_certificate/blob/main/1_data_collection.ipynb



Task 1: Request and parse the SpaceX launch data using the GET request

To make the requested JSON results more consistent, we will use the following static response object for this project:

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

We should see that the request was successfull with the 200 status response code

```
response.status_code
```

```
200
```

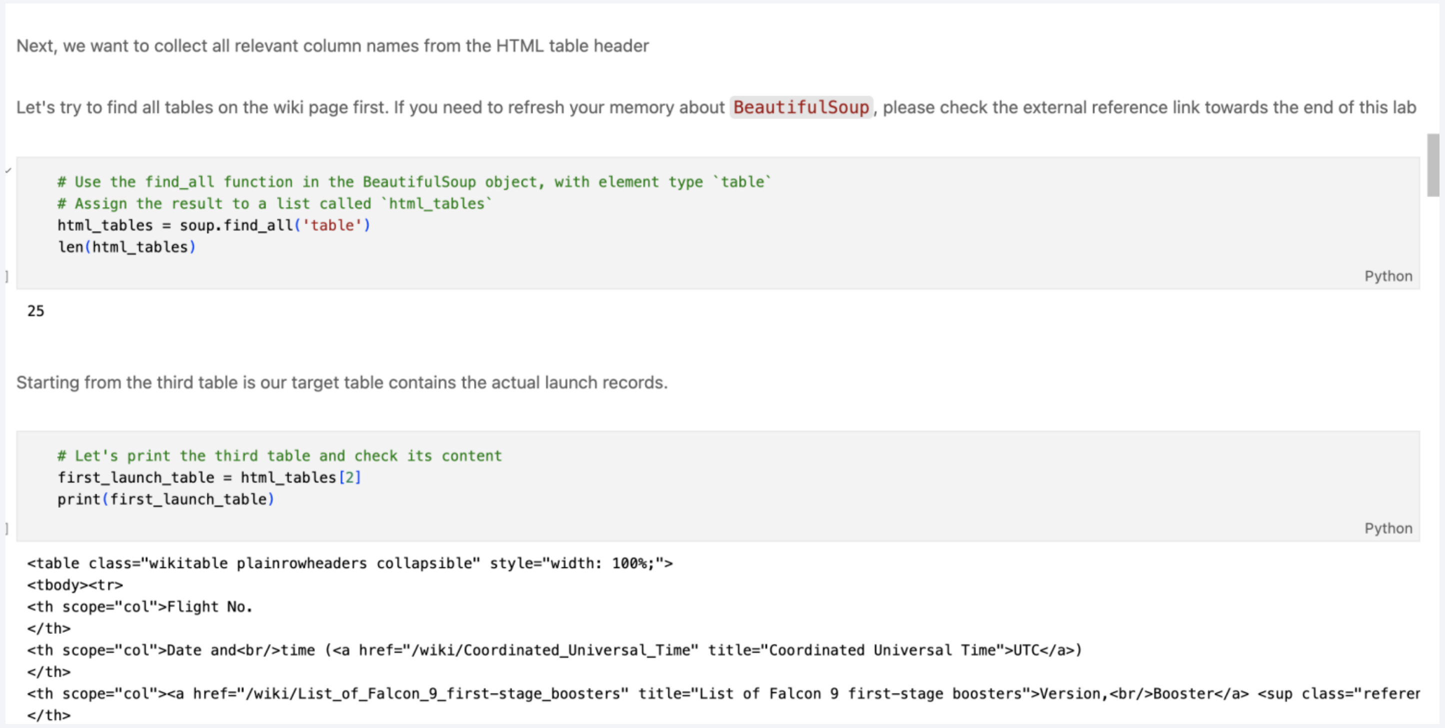
Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

```
data = pd.json_normalize(response.json())
```

Data Collection - Scraping

- Used web scrapping to collect Falcon 9 historical launch records from Wikipidea

- GitHub link: https://github.com/ArinaArtiukevich/ibm_ds_professional_certificate/blob/main/2_webscraping.ipynb


Next, we want to collect all relevant column names from the HTML table header

Let's try to find all tables on the wiki page first. If you need to refresh your memory about `BeautifulSoup`, please check the external reference link towards the end of this lab

```
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = soup.find_all('table')
len(html_tables)
```

Python

```
25
```

Starting from the third table is our target table contains the actual launch records.

```
# Let's print the third table and check its content
first_launch_table = html_tables[2]
print(first_launch_table)
```

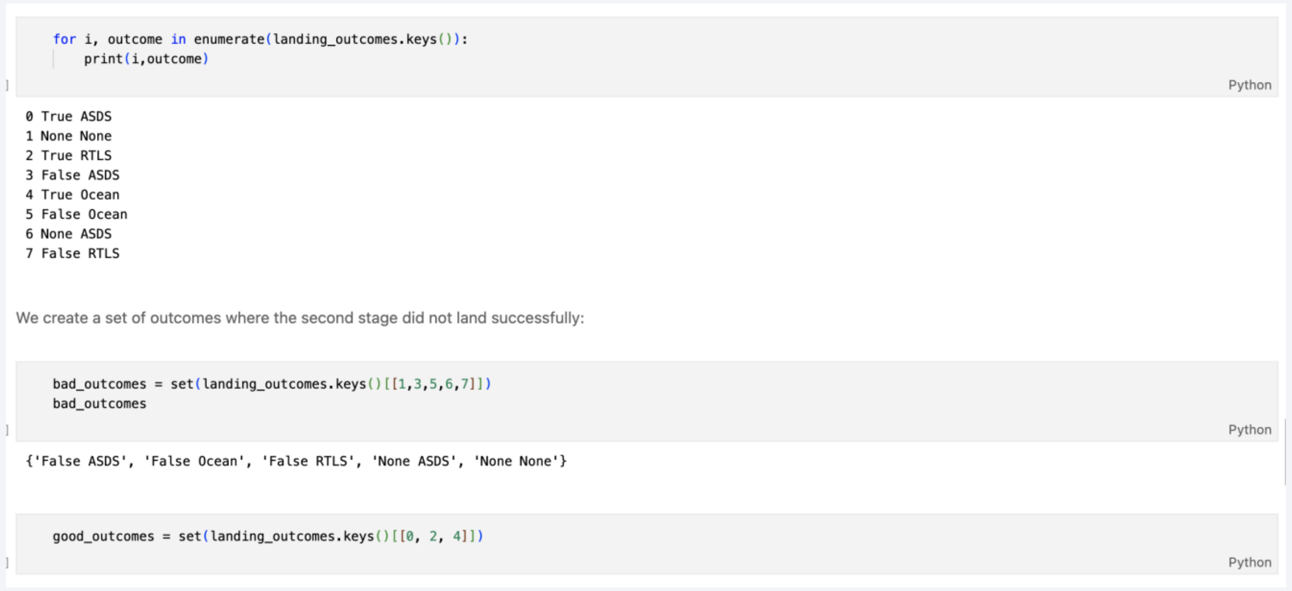
Python

```
<table class="wikitable plainrowheaders collapsible" style="width: 100%;">
<tbody><tr>
<th scope="col">Flight No.
</th>
<th scope="col">Date and<br/>time (<a href="/wiki/Coordinated_Universal_Time" title="Coordinated Universal Time">UTC</a>)
</th>
<th scope="col"><a href="/wiki/List_of_Falcon_9_first-stage_boosters" title="List of Falcon 9 first-stage boosters">Version,<br/>Booster</a> <sup class="referer
</th>
```

Data Wrangling

- Performed EDA and determined Training Labels

- GitHub link: https://github.com/ArinaArtiukevich/ibm_ds_professional_certificate/blob/main/3_data_wrangling.ipynb



```
for i, outcome in enumerate(landing_outcomes.keys()):
    print(i,outcome)
```

Python

```
0 True ASDS
1 None None
2 True RTLS
3 False ASDS
4 True Ocean
5 False Ocean
6 None ASDS
7 False RTLS
```

We create a set of outcomes where the second stage did not land successfully:

```
bad_outcomes = set(landing_outcomes.keys()[[1,3,5,6,7]])
bad_outcomes
```

Python

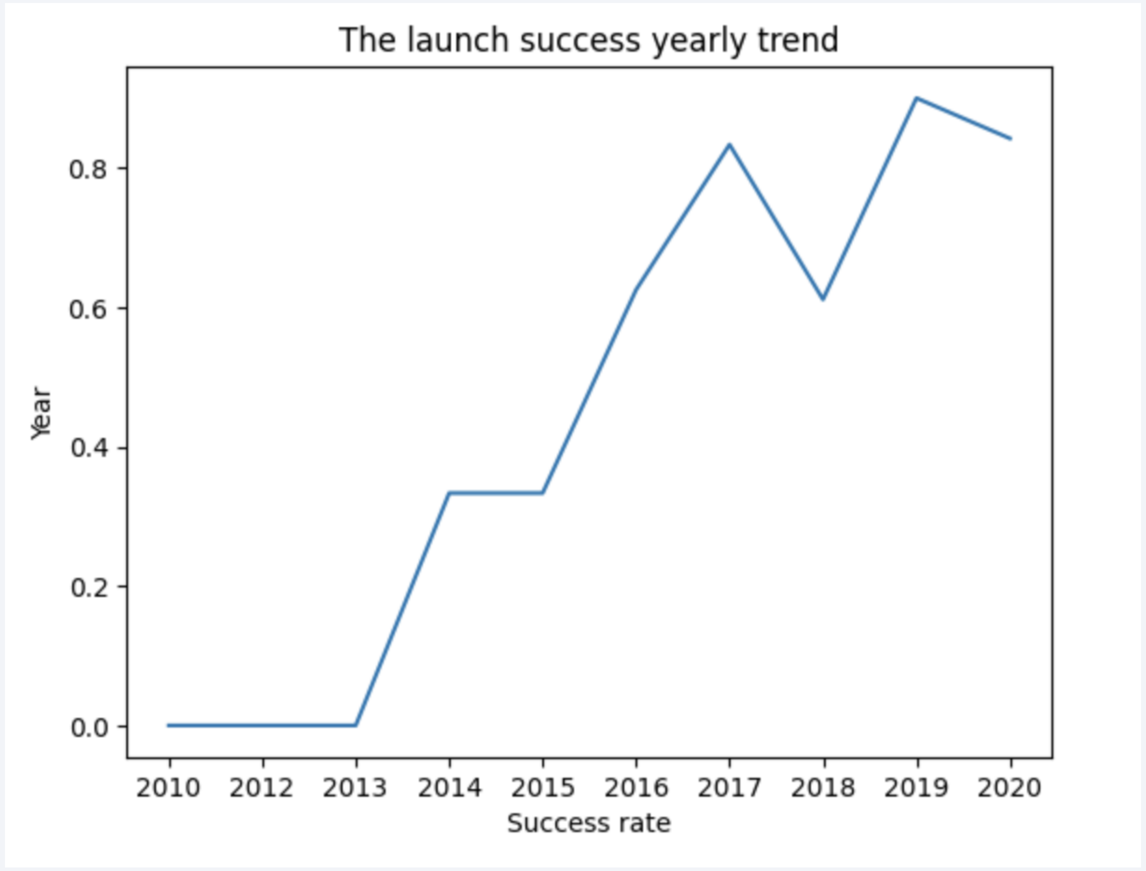
```
{'False ASDS', 'False Ocean', 'False RTLS', 'None ASDS', 'None None'}
```

```
good_outcomes = set(landing_outcomes.keys()[[0, 2, 4]])
```

Python

EDA with Data Visualization

- Performed EDA and Feature Engineering using Pandas and Matplotlib
- Found that the success rate since 2013 kept increasing till 2017 (stable in 2014) and after 2015 it started increasing
- GitHub link: https://github.com/ArinaArtiukevich/ibm_ds_professional_certificate/blob/main/5_eda_dataviz.ipynb



EDA with SQL

- Used SQLite to:
  - Displayed the names of the unique launch sites in the space mission
  - Displayed 5 records where launch sites begin with the string 'CCA'
  - Displayed the total payload mass carried by boosters launched by NASA (CRS)
  - Displayed average payload mass carried by booster version F9 v1.1
  - Listed the date when the first succesful landing outcome in ground pad was acheived.
  - Listed the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
  - Listed the total number of successful and failure mission outcomes
  - Ranked the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order.
- GitHub link: https://github.com/ArinaArtiukevich/ibm_ds_professional_certificate/blob/main/4_eda_sql.ipynb

Build an Interactive Map with Folium

- Marked all launch sites on a map
- Marked the success/failed launches for each site on the map
- Calculated the distances between a launch site to its proximities
- GitHub link: https://github.com/ArinaArtiukevich/ibm_ds_professional_certificate/blob/main/6_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

- Added a Launch Site Drop-down Input Component
- Added a callback function to render success-pie-chart based on selected site dropdown
- Added a Range Slider to Select Payload
- Added success-payload-scatter-chart scatter plot
- GitHub link: https://github.com/ArinaArtiukevich/ibm_ds_professional_certificate/blob/main/7_dash_app.py

Predictive Analysis (Classification)

- Performed EDA and determined Training Labels

o Created a column for the class

o Standardized the data

- Splitted into training data and test data
- Found best Hyperparameter for SVM, Classification Trees and Logistic Regression
- Found the method performs best using test data
- GitHub link: https://github.com/ArinaArtiukevich/ibm_ds_professional_certificate/blob/main/8_machine_learning.ipynb

Results

- Launch success has improved over given time,
- Most launch sites are close to equator,
- Decision Tree performed the best on the dataset.

Thank you!