Winning Space Race with Data Science

Yogesh Kissoondary
August 2022

Outline

- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion
- Appendix

Executive Summary

- **Summary of methodologies**
  - Data Collection using API and Web Scraping
  - Exploratory Data Analysis using SQL
  - Exploratory Data Analysis using Data Visualization
  - Interactive Visual Analytics with Folium
  - Interactive Visual Analytics with Dashboard
  - Machine Learning Prediction
- **Summary of all results**
  - Exploratory Data Analysis
  - Interactive analytics in Dashboard
  - Predictive Analytics using different algorithms

Introduction

- Project background and context

  Space X 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 Space X can reuse the first stage. Therefore, if we can determine if the first stage will land, we can determine the cost of a launch. The goal of the project is to create a machine learning pipeline to predict if the first stage will land successfully.

- Problems you want to find answers
  - The factors affect the successful landing of the rocket
  - The interaction of various features that determine the success rate of a successful landing.
  - The operating conditions assist the successful landing

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using SpaceX API and web scraping from Wikipedia
  - Perform data wrangling
  - Clean data to be suitable for exploratory data analysis and machine learning.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

- The data was collected using various methods
  - Data was collected using get request to the SpaceX API.
  - using .json() and .json_normalize(), the response content is turned to a pandas dataframe.
  - The missing values are examined and replace with the appropriate values
  - A web scraping from Wikipedia for Falcon 9 launch records is performed with BeautifulSoup.
  - Using the BeautifulSoup, the HTML tables are parsed and converted to pandas dataframe.

Data Collection – SpaceX API

- get request to the SpaceX API is used to collect data, clean the requested data and did some data wrangling and formatting
- https://github.com/yogeshsurf/IBM-Data-Science-Capstone/blob/59e95ac43ab100e44bad4953a73e9e3b71cfe6f7/Collecting%20the%20data.ipynb

```
In [10]: # 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_

In [11]: response.status_code

Out[11]: 200

In [12]: # Use json_normalize meethod to convert the json result into a dataframe
         response.json()
         data = pd.json_normalize(response.json())

In [13]: # Get the head of the dataframe
         data.head()
```

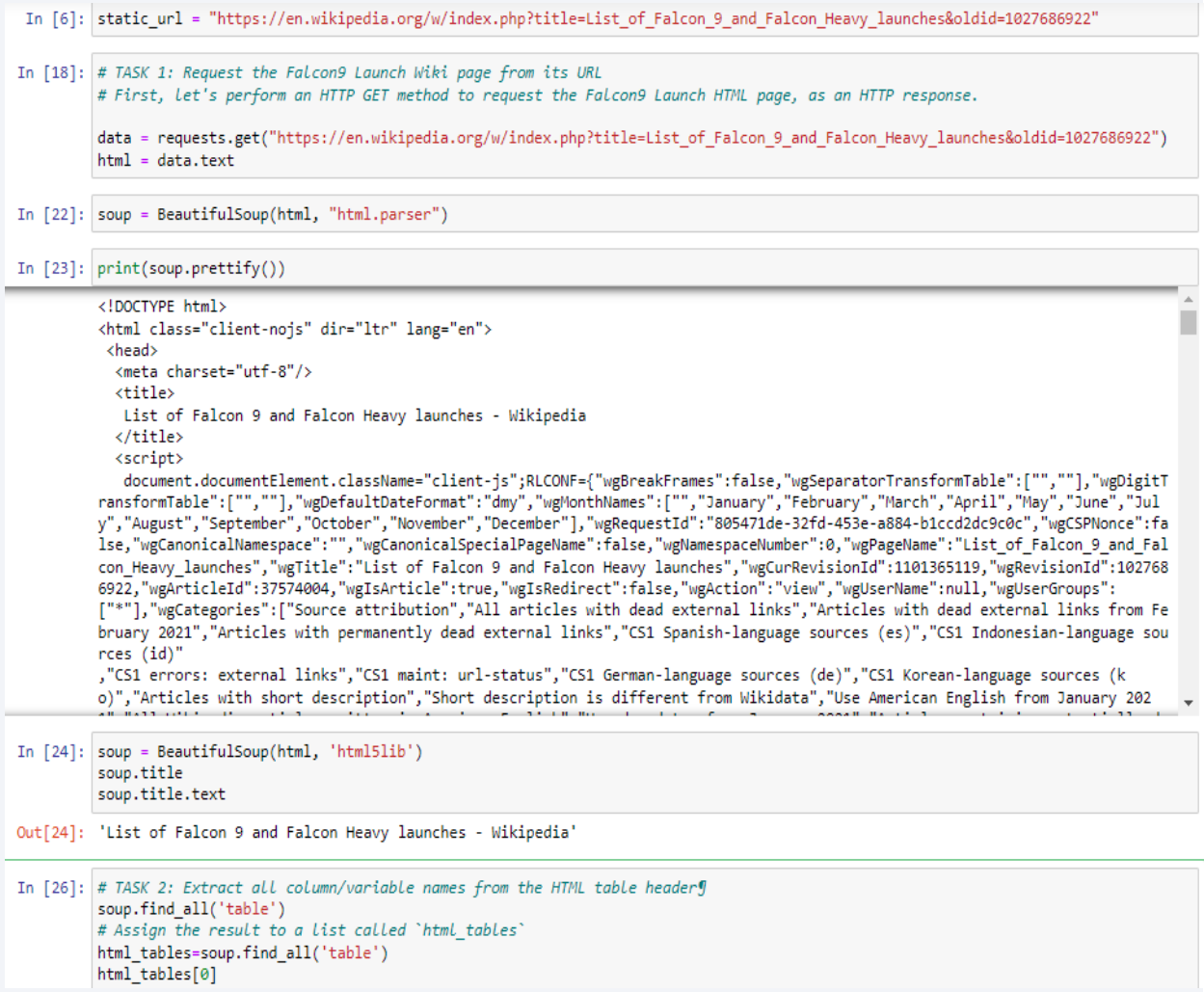
Out[13]:

	static_fire_date_utc	static_fire_date_unix	net	window	rocket	success	failures	details	crew	ships	capsules	
0	2006-03-17T00:00:00.000Z	1.142554e+09	False	0.0	5e9d0d95eda69955f709d1eb	False	[{'time': 33, 'altitude': None, 'reason': 'merlin engine failure'}]	Engine failure at 33 seconds and loss of vehicle	[]	[]	[]	[5eb0e4b5b6c3bl
1	None	NaN	False	0.0	5e9d0d95eda69955f709d1eb	False	[{'time': 301, 'altitude': 289, 'reason': 'harmonic oscillation leading to premature engine shutdown'}]	Successful first stage burn and transition to second stage, maximum altitude 289 km, Premature engine shutdown at T+7 min 30 s, Failed to reach orbit, Failed to recover first stage	[]	[]	[]	[5eb0e4b6b6c3bl
2	None	NaN	False	0.0	5e9d0d95eda69955f709d1eb	False	[{'time': 140, 'altitude': 35, 'reason': 'residual stage-1 thrust led to collision between stage 1 and stage	Residual stage 1 thrust led to collision between stage 1 and stage 2	[]	[]	[]	[5eb0e4b6b6c3bl 5eb0e4b6b6c3bl

Data Collection - Scraping

- A web scraping from Wikipedia for Falcon 9 launch records is performed with BeautifulSoup.
- Using the BeautifulSoup, the HTML tables are parsed and converted to pandas dataframe.

https://github.com/yogeshsurf/IBM-Data-Science-Capstone/blob/59e95ac43ab100e44bad4953a73e9e3b71cfe6f7/Web%20scraping%20Falcon%209%20and%20Falcon%20Heavy%20Launches%20Records%20from%20Wikipedia.ipynb

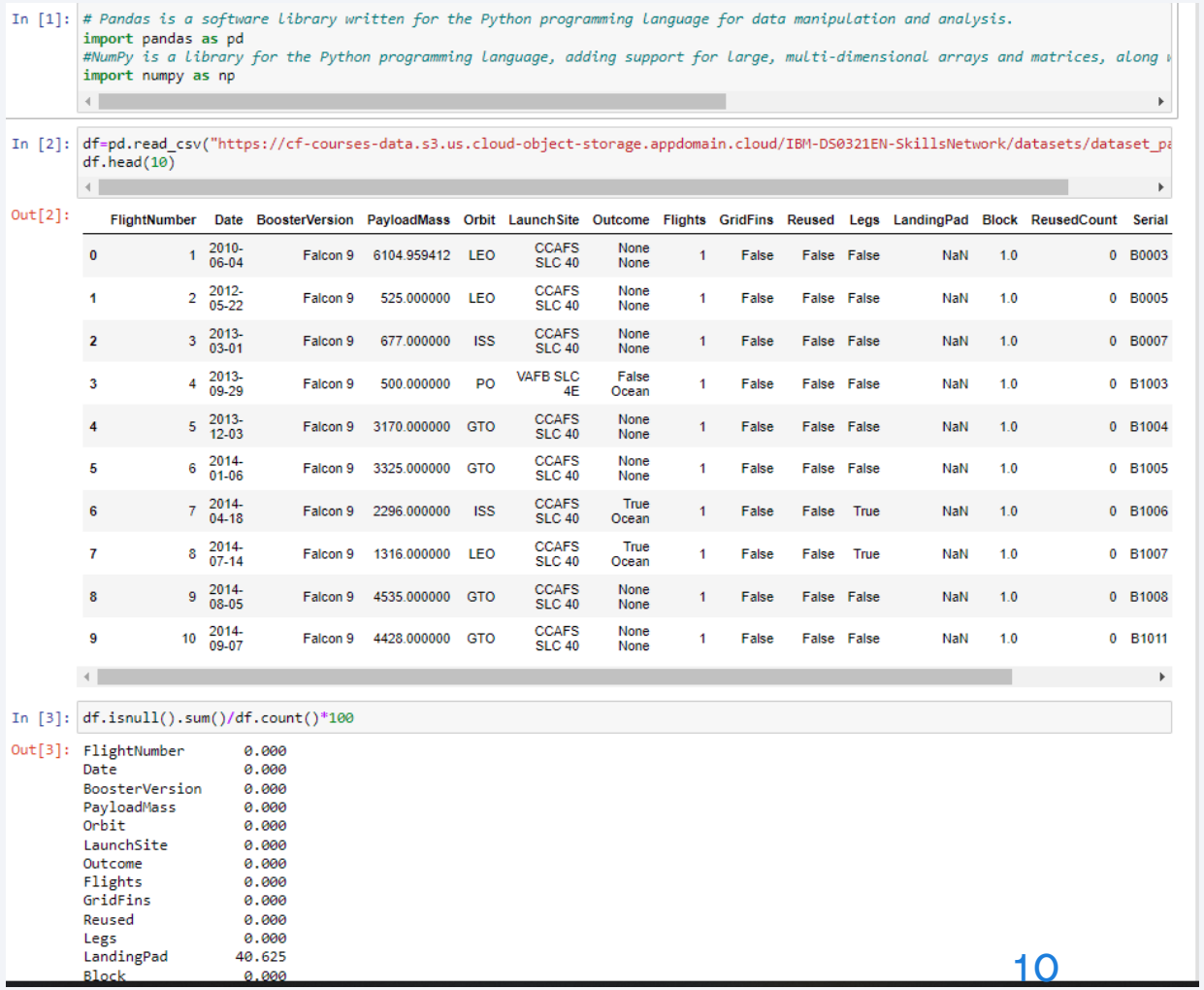


```
In [6]: static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"

In [18]: # TASK 1: Request the Falcon9 Launch Wiki page from its URL
         # First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

         data = requests.get("https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922")
         html = data.text

In [22]: soup = BeautifulSoup(html, "html.parser")

In [23]: print(soup.prettify())

<!DOCTYPE html>
<html class="client-nojs" dir="ltr" lang="en">
 <head>
  <meta charset="utf-8"/>
  <title>
   List of Falcon 9 and Falcon Heavy launches - Wikipedia
  </title>
  <script>
   document.documentElement.className="client-js";RLCONF={"wgBreakFrames":false,"wgSeparatorTransformTable":["",""],"wgDigitT
ransformTable":["",""],"wgDefaultDateFormat":"dmy","wgMonthNames":["","January","February","March","April","May","June","Jul
y","August","September","October","November","December"],"wgRequestId":"805471de-32fd-453e-a884-b1ccd2dc9c0c","wgCSPNonce":fa
lse,"wgCanonicalNamespace":"","wgCanonicalSpecialPageName":false,"wgNamespaceNumber":0,"wgPageName":"List_of_Falcon_9_and_Fal
con_Heavy_launches","wgTitle":"List of Falcon 9 and Falcon Heavy launches","wgCurRevisionId":1101365119,"wgRevisionId":102768
6922,"wgArticleId":37574004,"wgIsArticle":true,"wgIsRedirect":false,"wgAction":"view","wgUserName":null,"wgUserGroups":
["*"],"wgCategories":["Source attribution","All articles with dead external links","Articles with dead external links from Fe
bruary 2021","Articles with permanently dead external links","CS1 Spanish-language sources (es)","CS1 Indonesian-language sou
rces (id)"
,"CS1 errors: external links","CS1 maint: url-status","CS1 German-language sources (de)","CS1 Korean-language sources (k
o)","Articles with short description","Short description is different from Wikidata","Use American English from January 202

In [24]: soup = BeautifulSoup(html, 'html5lib')
         soup.title
         soup.title.text

Out[24]: 'List of Falcon 9 and Falcon Heavy launches - Wikipedia'

In [26]: # TASK 2: Extract all column/variable names from the HTML table header¶
         soup.find_all('table')
         # Assign the result to a list called `html_tables`
         html_tables=soup.find_all('table')
         html_tables[0]
```

Data Wrangling

- Using the Pandas and Numpy libraries:
- The data is cleaned and duplication and missing values are examined
- Missing values are replaced with the appropriate values

https://github.com/yogeshsurf/IBM-Data-Science-Capstone/blob/59e95ac43ab100e44bad4953a73e9e3b71cfe6f7/Data%20wrangling.ipynb



In [1]:
```
# 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
import numpy as np
```

In [2]:
```
df=pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/dataset_p
df.head(10)
```

Out[2]:

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0007
3	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	NaN	1.0	0	B1003
4	5	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1004
5	6	2014-01-06	Falcon 9	3325.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1005
6	7	2014-04-18	Falcon 9	2296.000000	ISS	CCAFS SLC 40	True Ocean	1	False	False	True	NaN	1.0	0	B1006
7	8	2014-07-14	Falcon 9	1316.000000	LEO	CCAFS SLC 40	True Ocean	1	False	False	True	NaN	1.0	0	B1007
8	9	2014-08-05	Falcon 9	4535.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1008
9	10	2014-09-07	Falcon 9	4428.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1011

In [3]:
```
df.isnull().sum()/df.count()*100
```

Out[3]:
```
FlightNumber      0.000
Date              0.000
BoosterVersion    0.000
PayloadMass       0.000
Orbit             0.000
LaunchSite        0.000
Outcome           0.000
Flights           0.000
GridFins          0.000
Reused            0.000
Legs              0.000
LandingPad       40.625
Block             0.000
```

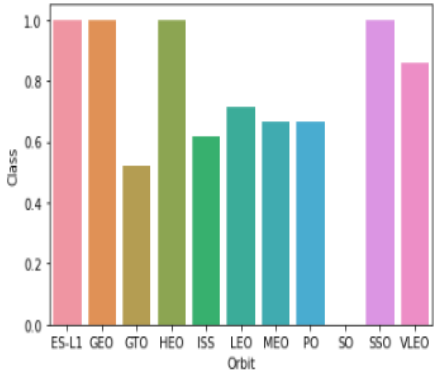
EDA with Data Visualization

- The data is explored by visualizing the relationship between flight number and launch Site, payload and launch site, success rate of each orbit type, flight number and orbit type, the launch success yearly

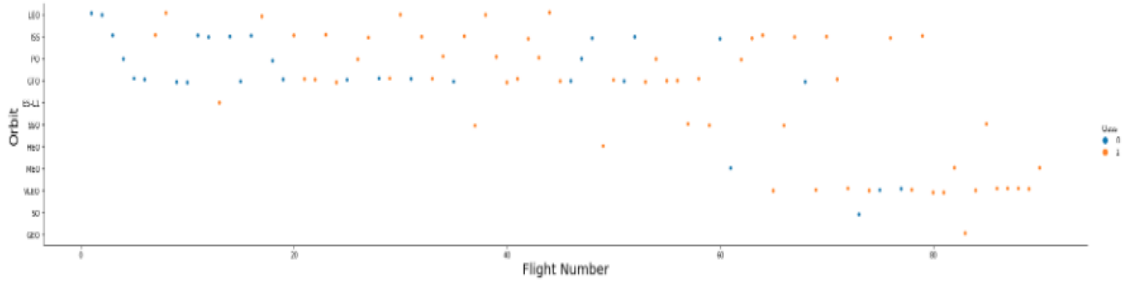
https://github.com/yogeshsurf/IBM-Data-Science-Capstone/blob/888d3535a92af0062d5fd7777e18ebbf14d708ff/Exploring%20and%20Preparing%20Data.ipynb

In [8]:
```
# TASK 3: Visualize the relationship between success rate of each orbit type

# HINT use groupby method on Orbit column and get the mean of Class column

ddf=df.groupby('Orbit').mean()[['Class']].reset_index()
sns.barplot(y=ddf['Class'], x=ddf['Orbit'],  data=df)
```

Out[8]: <AxesSubplot:xlabel='Orbit', ylabel='Class'>



In [9]:
```
# TASK 4: Visualize the relationship between FlightNumber and Orbit type

# Plot a scatter point chart with x axis to be FlightNumber and y axis to be the Orbit, and hue to be the class value

sns.catplot(y="Orbit", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



EDA with SQL

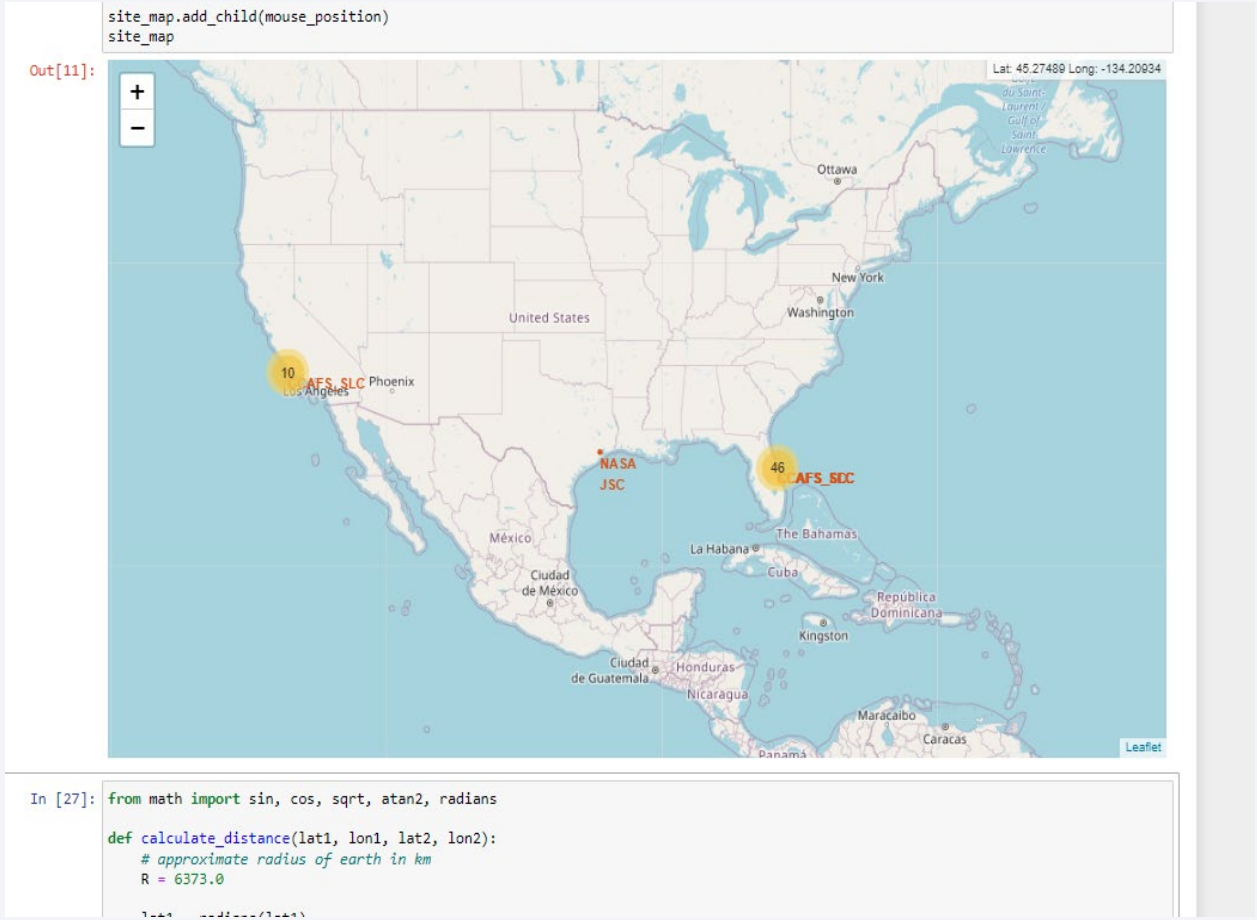
- The SpaceX dataset is used with MYSQL database
- Different queries are examined using SQL magic function in Jupyter.
- The queries include:
  - The names of unique launch sites in the space mission.
  - The total payload mass carried by boosters launched by NASA (CRS)
  - The average payload mass carried by booster version F9 v1.1
  - The total number of successful and failure mission outcomes
  - The failed landing outcomes in drone ship, their booster version and launch site names.

https://github.com/yogeshsurf/IBM-Data-Science-Capstone/blob/b191e0b795a878cfe76a71eb8fdf44f8e22722b9/SQL%20Notebook%20for%20Peer%20Assignment.ipynb

Build an Interactive Map with Folium

- All launch sites are explained on map with adding objects such as markers, circles, lines to mark the success or failure of launches for each site.
- An assignment of failure or success on each site is differentiate by color .
- The distances between a launch site to its proximities to railway, high-way and coasts are explained.

https://github.com/yogeshsurf/IBM-Data-Science-Capstone/blob/61b288474ad4d1d12781cc97179bad572403f26a/Launch%20Sites%20Locations%20Analysis%20with%20Folium.ipynb



Build a Dashboard with Plotly Dash

- a Plotly Dash application for users to perform interactive visual analytics on SpaceX launch data in real-time.
- This dashboard application contains input components such as a dropdown list and a range slider to interact with a pie chart and a scatter point chart.

https://github.com/yogeshsurf/IBM-Data-Science-Capstone/blob/84b8c723ae11c784d1d20e3b8857318cb2f54468/Dashboard%20Application%20with%20Plotly%20Dash.ipynb

Predictive Analysis (Classification)

- The predictive analysis is started by importing the required packages
- Indicating the input data and the output data
- Splitting the data into train and test data
- Try multiple algorithm to get the best results for the prediction
- These algorithms are :
    - Logistic Regression classification algorithm
    - Support Vector Machine classification algorithm
    - Decision Tree classification algorithm
    - K Nearest Neighbors classification algorithm

https://github.com/yogeshsurf/IBM-Data-Science-Capstone/blob/082da0938a51278f3be03dac0de3570afa3152f6/Machine%20Learning%20Prediction.ipynb

Results

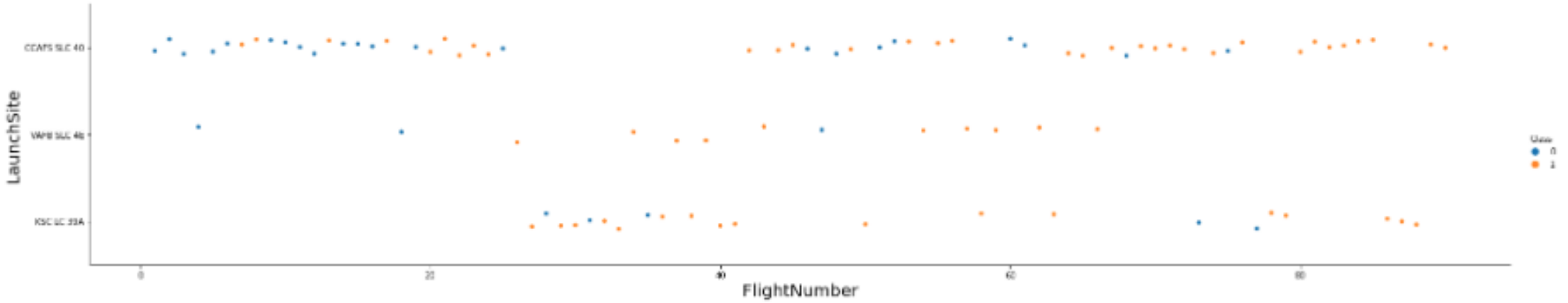
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

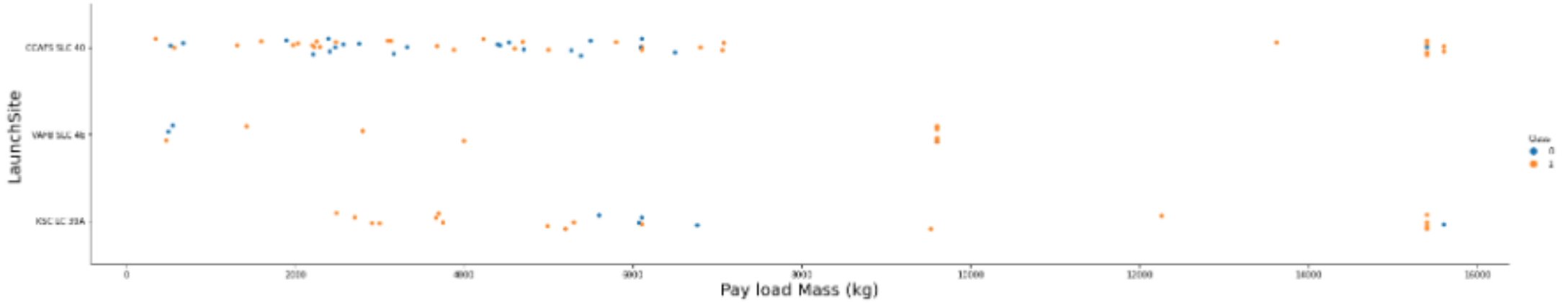
Flight Number vs. Launch Site

In [6]:
```
# Task 1 Visualize the relationship between Flight Number and Launch Site

# Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and hue to be the class value

sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("FlightNumber",fontsize=20)
plt.ylabel("LaunchSite",fontsize=20)
plt.show()
```



Payload vs. Launch Site

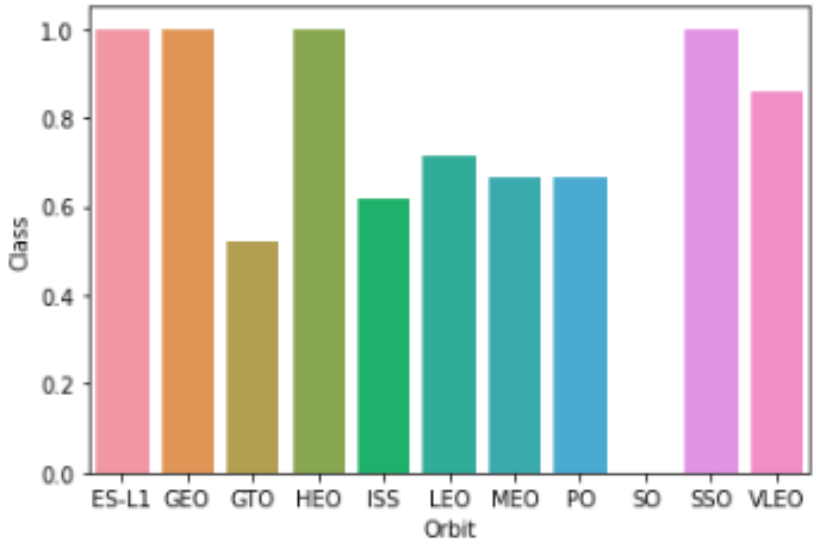
In [7]:
```
# TASK 2: Visualize the relationship between Payload and Launch Site

# Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the launch site, and hue to be the class value

sns.catplot(y="LaunchSite", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.xlabel("Pay load Mass (kg)",fontsize=20)
plt.ylabel("LaunchSite",fontsize=20)
plt.show()
```



Success Rate vs. Orbit Type

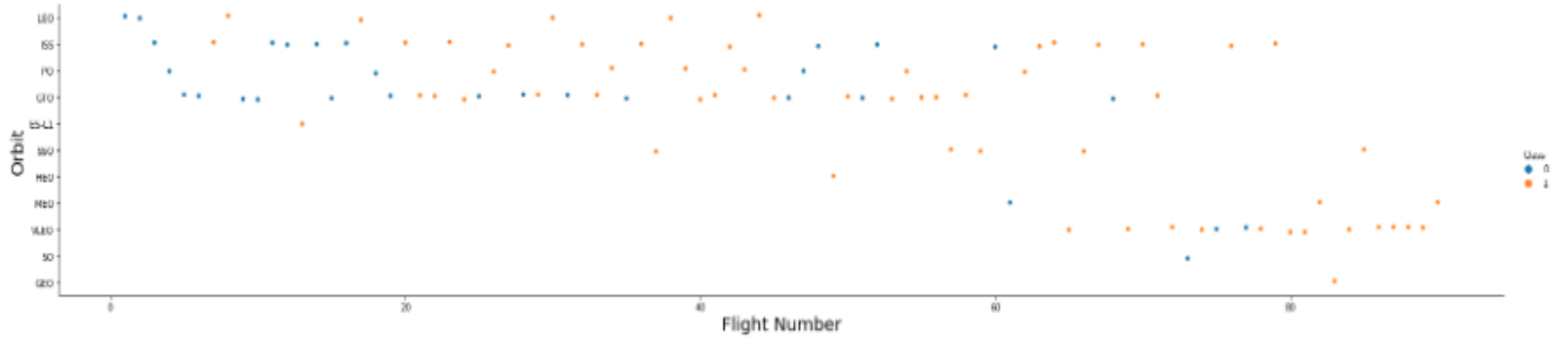
In [8]:
```
# TASK 3: Visualize the relationship between success rate of each orbit type

# HINT use groupby method on Orbit column and get the mean of Class column

ddf=df.groupby('Orbit').mean()[['Class']].reset_index()
sns.barplot(y=ddf['Class'], x=ddf['Orbit'],  data=df)
```

Out[8]:
```
<AxesSubplot:xlabel='Orbit', ylabel='Class'>
```



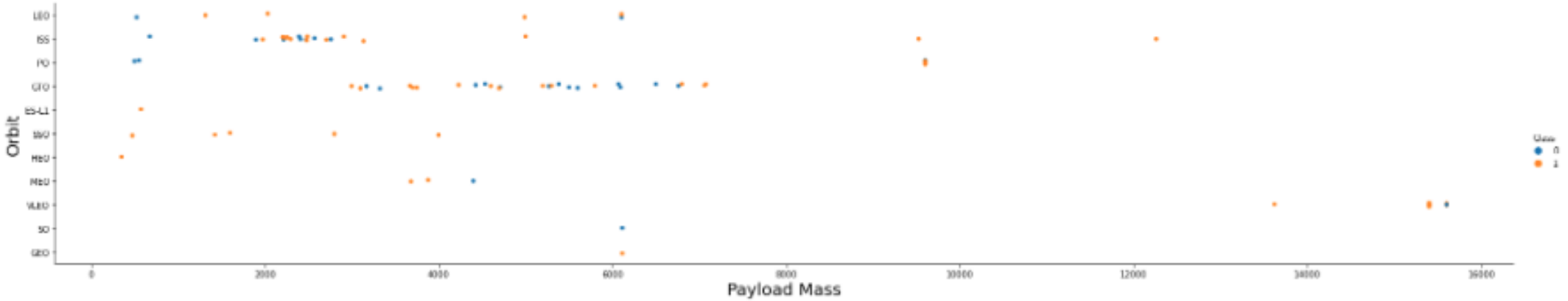
Flight Number vs. Orbit Type

In [9]:
```
# TASK 4: Visualize the relationship between FlightNumber and Orbit type

# Plot a scatter point chart with x axis to be FlightNumber and y axis to be the Orbit, and hue to be the class value

sns.catplot(y="Orbit", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



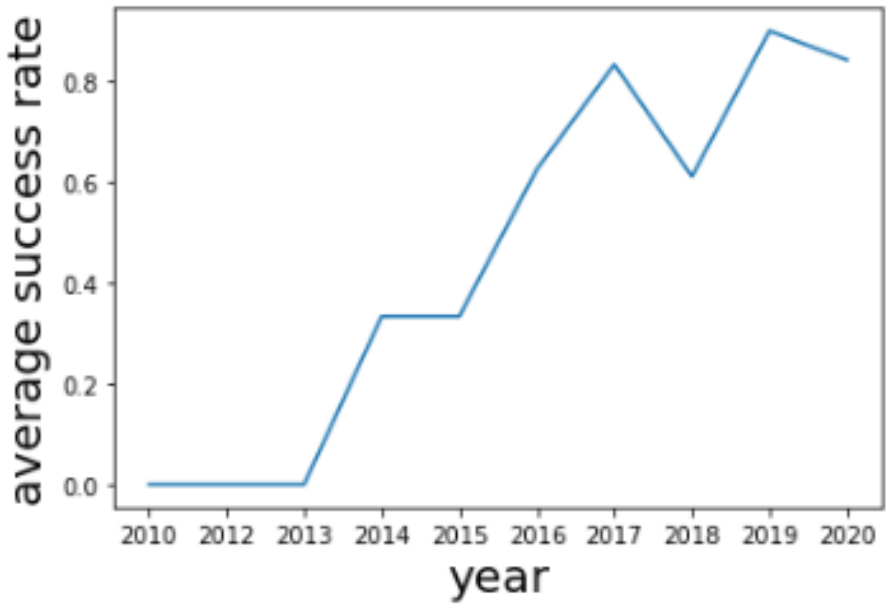
Payload vs. Orbit Type

In [10]:
```
# TASK 5: Visualize the relationship between Payload and Orbit type

# Plot a scatter point chart with x axis to be Payload and y axis to be the Orbit, and hue to be the class value

sns.catplot(y="Orbit", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.xlabel("Payload Mass",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



Launch Success Yearly Trend

```
# Plot a line chart with x axis to be the extracted year and y axis to be the success rate
sns.lineplot(y="Class", x="year", data=dff1)
plt.xlabel("year",fontsize=20)
plt.ylabel("average success rate",fontsize=20)
plt.show()
```



All Launch Site Names

click to expand output; double click to hide output

```
# Display the names of the unique launch sites in the space mission

%sql select DISTINCT Launch_Site from SPACEX
```

```
 * ibm_db_sa://dpj01030:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30756/bludb;sec
urity=SSL;
Done.
```

Out[6]:

launch_site
CCAFS LC-40
CCAFS SLC-40
KSC LC-39A
VAFB SLC-4E

Launch Site Names Begin with 'CCA'

In [7]:
```
# Task 2
# Display 5 records where launch sites begin with the string 'CCA'

%sql select * from SPACEX WHERE Launch_Site LIKE 'CCA%' limit 5
```

```
 * ibm_db_sa://dpj01030:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30756/bludb;security=SSL;
Done.
```

Out[7]:

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
None	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
None	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
None	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
None	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
None	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

In [8]: # Task 3

Total Payload Mass

- Calculate the total payload carried by boosters from NASA

[8]:
```
# Task 3

# Display the total payload mass carried by boosters launched by NASA (CRS)
%sql select SUM(payload_mass__kg_) from SPACEX where customer like '%NASA (CRS)%'
```

```
 * ibm_db_sa://dpj01030:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30756/bludb;security=SSL;
Done.
```

[8]:

1
48213

Average Payload Mass by F9 v1.1

In [30]:
```
# Task 4

# Display average payload mass carried by booster version F9 v1.1

%sql select avg(payload_mass__kg_) AS averagePayloadMass from SPACEX where booster_version like '%F9 v1.1%'
```

```
 * ibm_db_sa://dpj01030:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30756/bludb;sec
urity=SSL;
Done.
```

Out[30]:

averagepayloadmass
2534

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad
- Present your query result with a short explanation here

Successful Drone Ship Landing with Payload between 4000 and 6000

In [34]:
```
# Task 6
# List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

%sql select booster_version* from spacex where landing__outcome like 'Success (drone ship)%' and payload_mass__kg_ > 4000 and pay
```

```
 * ibm_db_sa://dpj01030:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30756/bludb;sec
urity=SSL;
Done.
```

Out[34]:

booster_version
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

In [12]:
```
# Task 7
# List the total number of successful and failure mission outcomes
%sql select mission_outcome, count (mission_outcome) from spacex group by mission_outcome
```

```
 * ibm_db_sa://dpj01030:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30756/bludb;security=SSL;
Done.
```

Out[12]:

mission_outcome	2
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

Boosters Carried Maximum Payload

In [24]:
```
# Task 8
# List the names of the booster_versions which have carried the maximum payload mass. Use a subquery¶


%sql select booster_version, payload_mass__kg_ from spacex where payload_mass__kg_ = (select max(payload_mass__kg_) from spacex)
```

```
 * ibm_db_sa://dpj01030:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30756/bludb;sec
urity=SSL;
Done.
```

Out[24]:

booster_version	payload_mass__kg_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600

2015 Launch Records

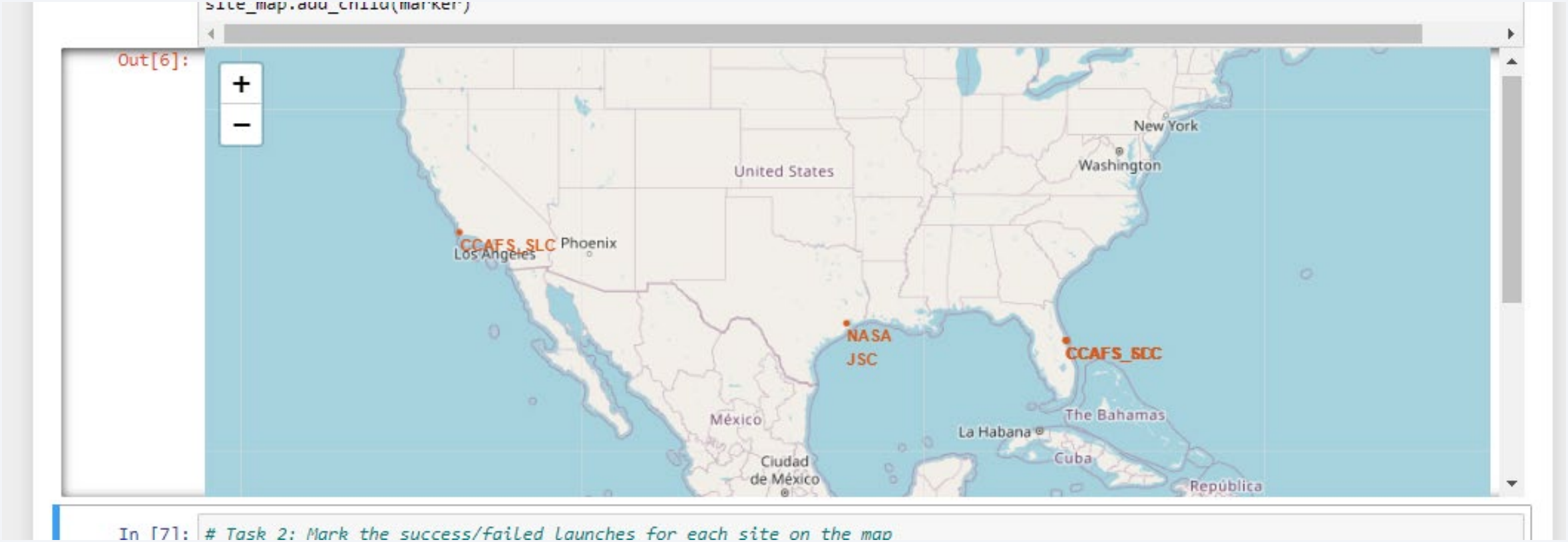
- The date format is showing as none when uploaded. I tried a few combination and unfortunately it is not working

Rank Landing Outcomes Between 2010-06-04 and 2017-03-20

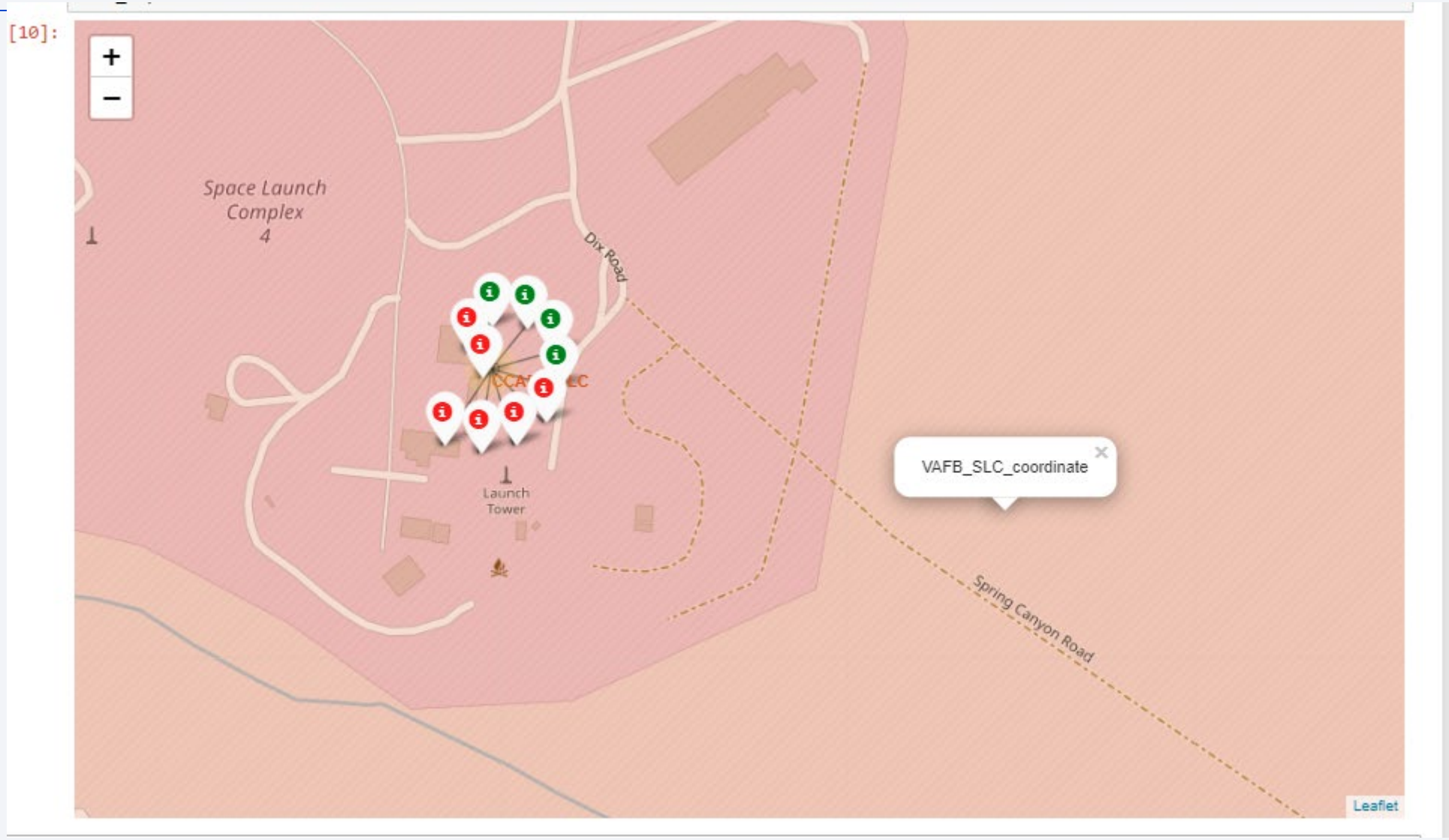
- The date format is showing as none when uploaded. I tried a few combination and unfortunately it is not working


Section 3
Launch Sites
Proximities Analysis

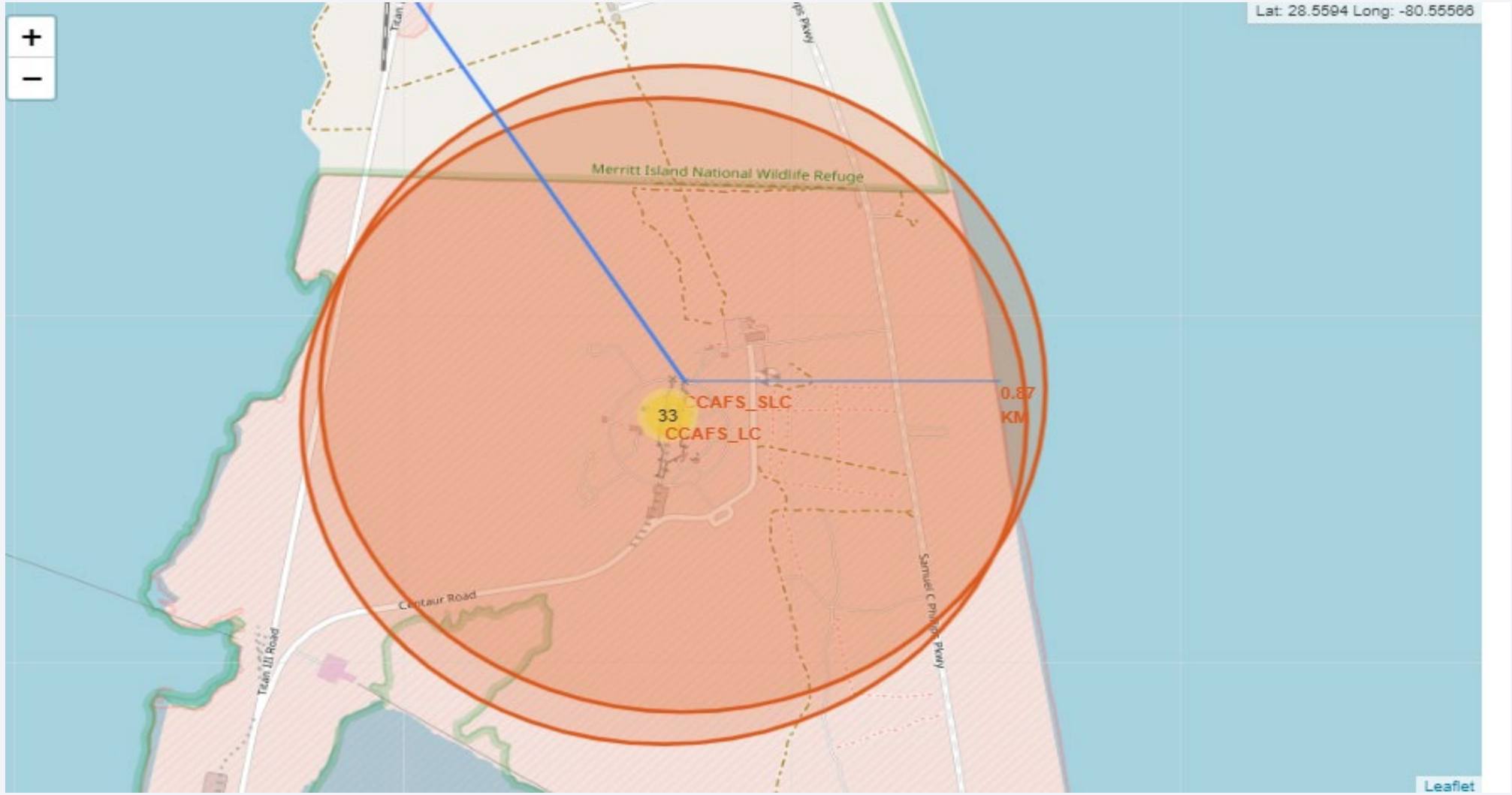
<Folium Map Screenshot 1>

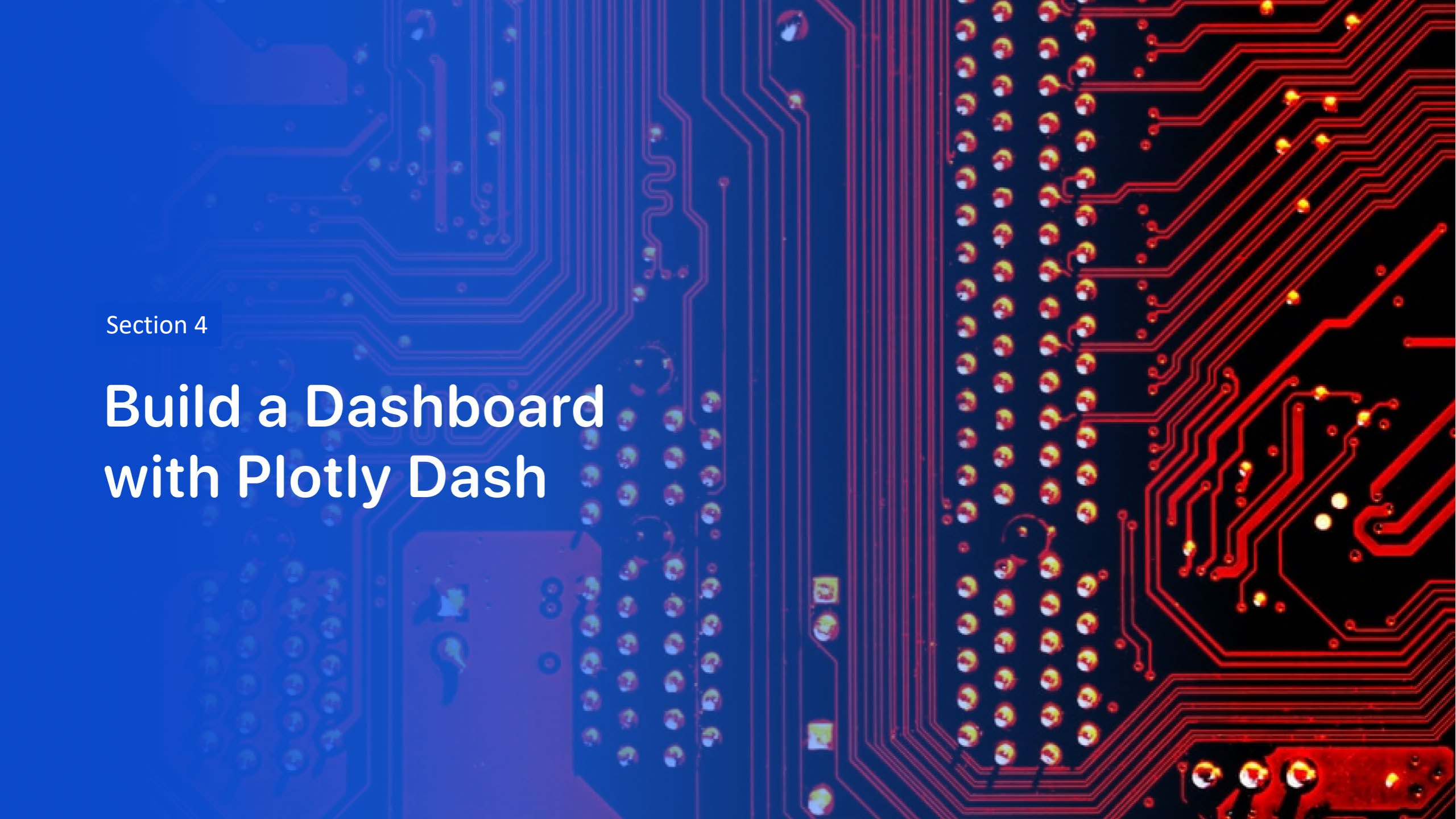


<Folium Map Screenshot 2>

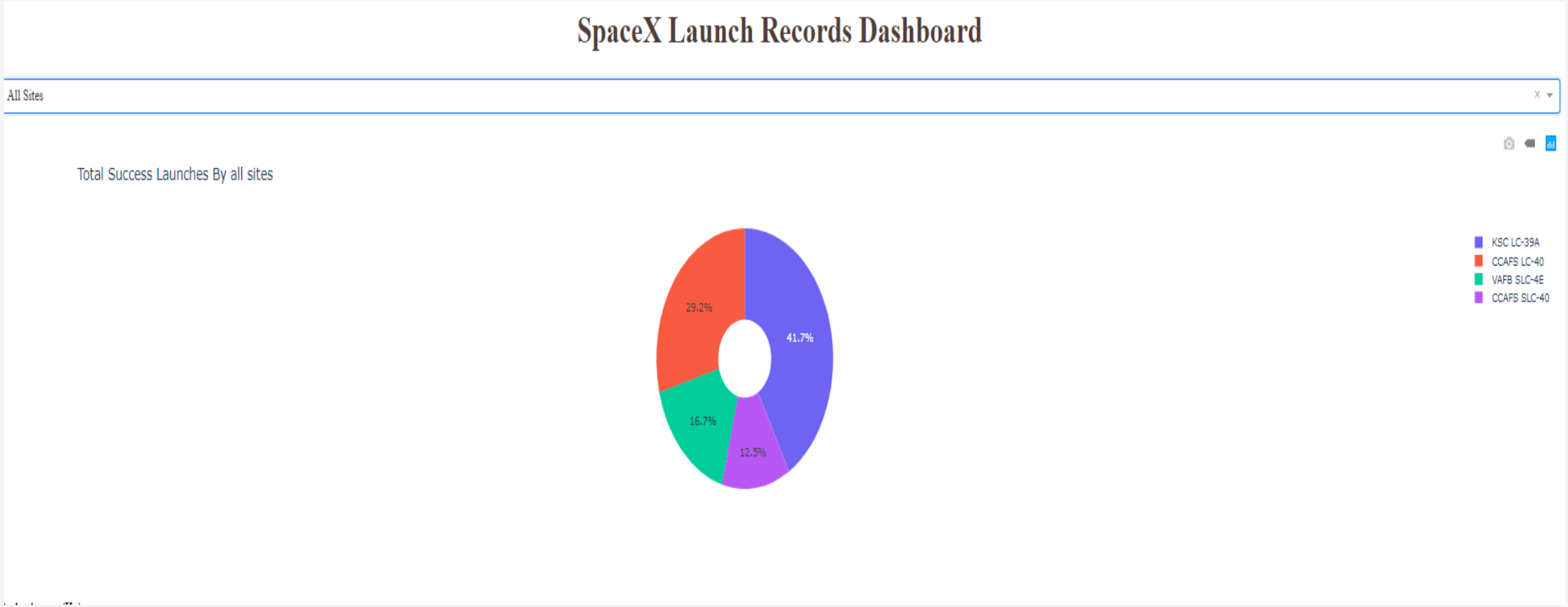


<Folium Map Screenshot 3>

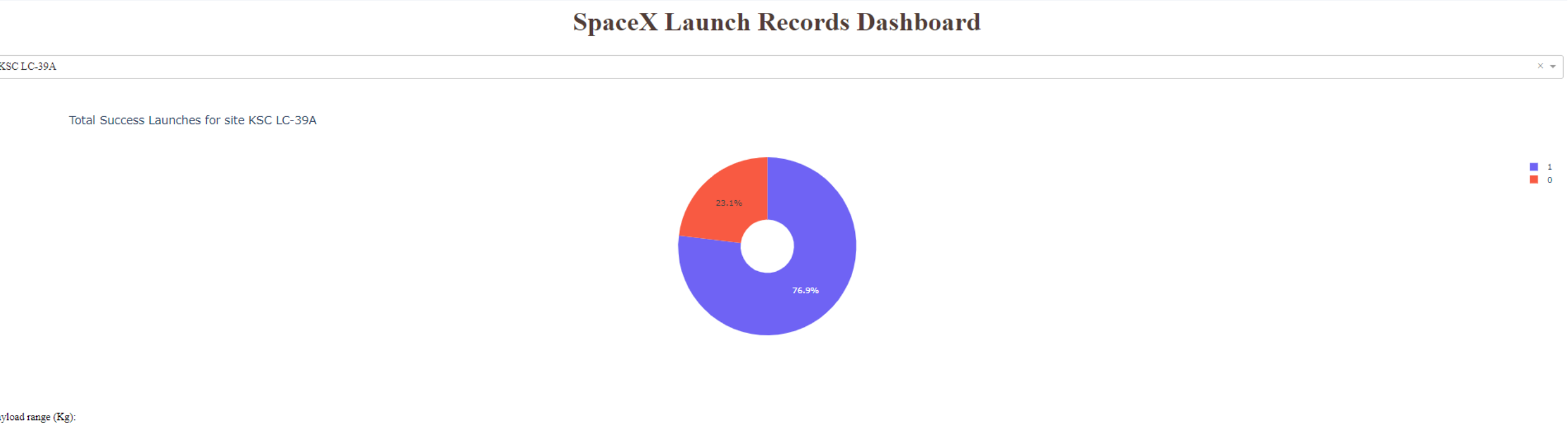



Section 4
Build a Dashboard with Plotly Dash

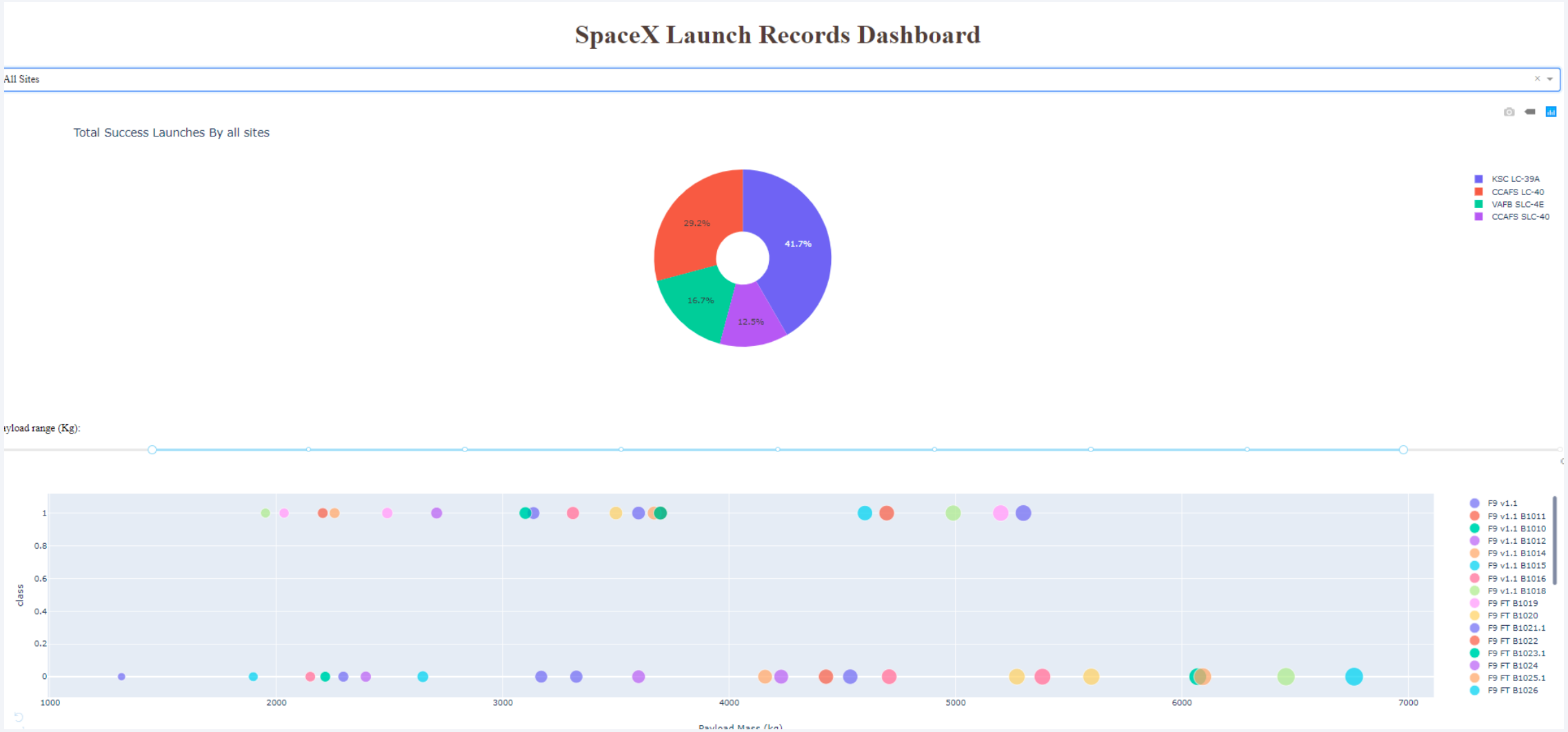
<Dashboard Screenshot 1>



<Dashboard Screenshot 2>



<Dashboard Screenshot 3>

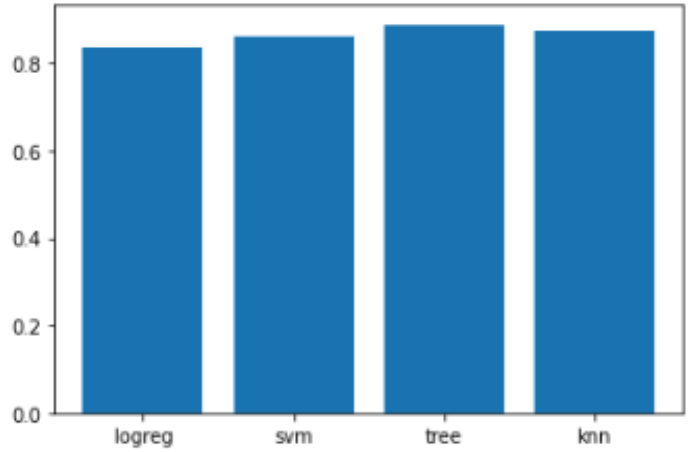


Section 5

Predictive Analysis (Classification)

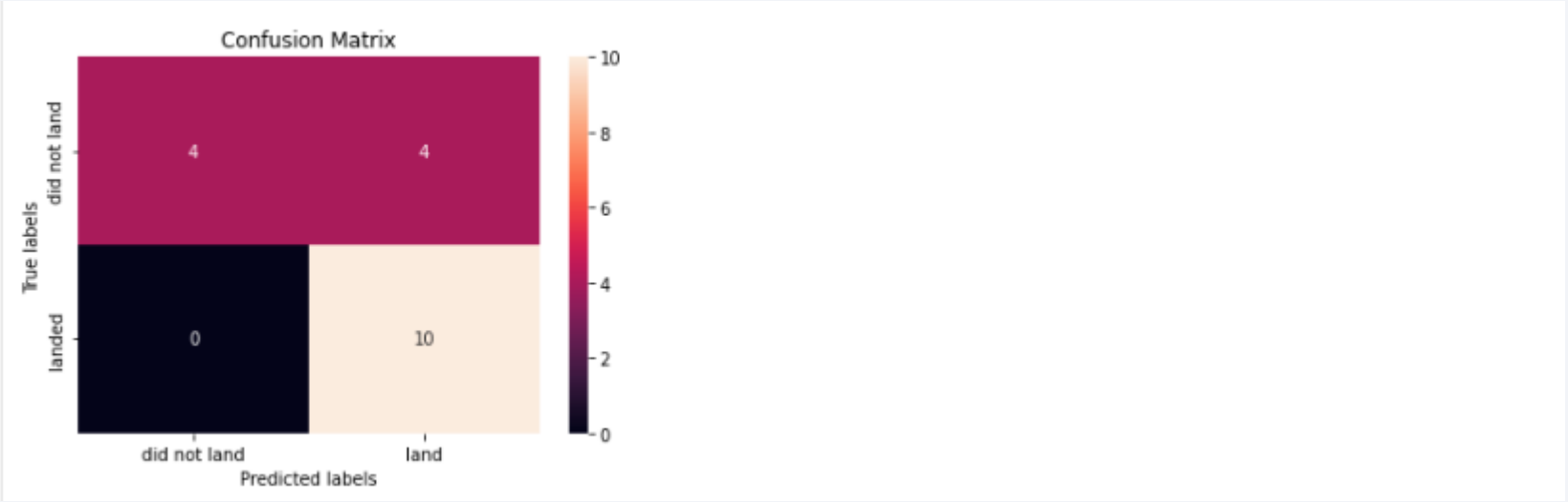
Classification Accuracy

In [69]:
```
# TASK 12

# Find the method performs

all_models={'logreg':logreg_cv.best_score_,
            'svm':svm_cv.best_score_,
            'tree':tree_cv.best_score_,
            'knn':knn_cv.best_score_}

import matplotlib.pyplot as plt
plt.bar(*zip(*all_models.items()))
plt.show()
```



It can be shown that the Decision tree has the best accuracy

Confusion Matrix



- the Decision tree has the best accuracy

Conclusions

- Launch success rate started to increase in 2013 till 2020.
- Orbits ES-L1, GEO, HEO and SSO have the most success rate.
- KSC LC-39A had the most successful launches of any sites.
- The Decision tree classifier is the best machine learning algorithm for this task.

Thank you!