IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Name>
<Date>

Kamalakbari77@gmail.com

Outline

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

Executive Summary

The Problem

- SpaceY wants to compete with SpaceX in sending spacecraft to space,
- Competition with SpaceX is not easy because it is the best company to send spacecraft to space inexpensively,
- The reason for more competency of SpaceX is its capability to reuse first-stage rockets,
- The main question of this study is: How can SpaceY compete with SpaceX?

The Solution

- To compete with SpaceX, we need to know them by applying Data Science methods,
- To capture and prepare data, we use methods such as SpaceX REST API and Web Scraping,
- Applying Exploratory Data Analysis and developing a Dashboard to explore patterns in the prepared data,
- Designing a Machine Learning pipeline for predicting the SpaceX first stage rockets.

The Results

- Correlations between different parameters can be observed in EDA and the designed Dashboard,
- Based on the provided parameters for missions with Falcon 9 rockets, such as payload mass and orbit, we can predict the success of a mission with 83% accuracy,
- SpaceY can apply the prediction results as a proxy to predict the cost of a mission,
- The captured results provide insights for SpaceY to find solutions to compete with SpaceX.

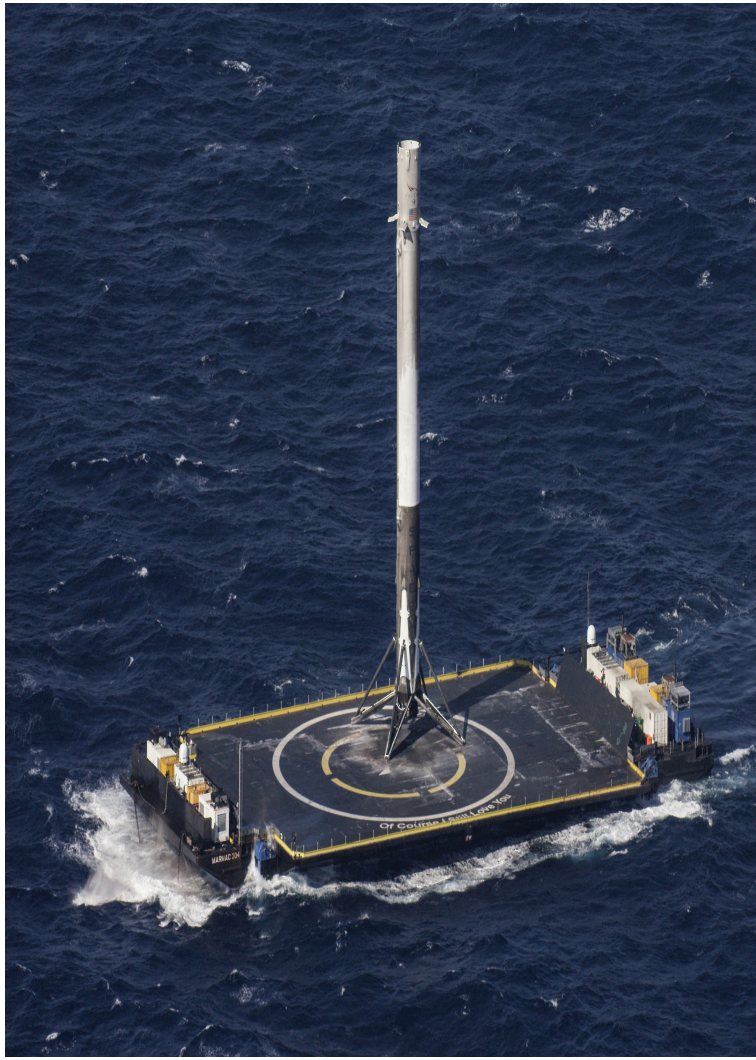
Challenges

- Capturing trusted datasets is challenging,
- This is a complicated problem, and the idea of SpaceY should be consulted and evaluated by an interdisciplinary team of experts in related fields

Executive Summary

- SpaceX is the best company to send spacecraft to space inexpensively
- The reason for more competency of SpaceX is its capability to reuse first-stage rockets
- The cost of a launch depends on landing the first stage

- SpaceX is the best company to send spacecraft to space inexpensively
- The reason for more competency of SpaceX is its capability to reuse first-stage rockets
- The cost of a launch depends on landing the first stage



https://en.wikipedia.org/wiki/File:CRS-8_(26239020092).jpg

Introduction

- SpaceX is the best company to send spacecraft to space inexpensively
- The reason for more competency of SpaceX is its capability to reuse first-stage rockets
- The cost of a launch depends on landing the first stage
- In SpaceY company, we want to know: How can we compete with SpaceX?
- By modelling the probability of a successful mission with provided variables, we can approximate the cost of a mission

Section 1

Methodology

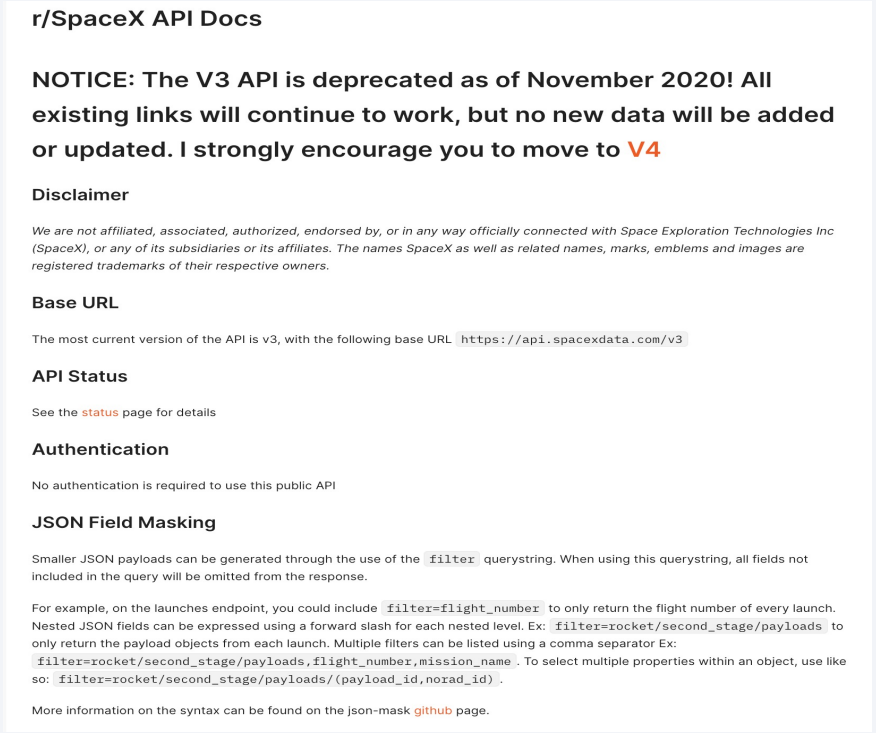
Kamalakbari77@gmail.com

Methodology

- Data collection
- Perform data wrangling
- Exploratory Data Analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Predictive analysis using classification models

Data Collection

- Data collection with REST API for SpaceX
- Data Collection with Web Scraping of Wikipedia Page



r/SpaceX API Docs

NOTICE: The V3 API is deprecated as of November 2020! All existing links will continue to work, but no new data will be added or updated. I strongly encourage you to move to V4

Disclaimer

We are not affiliated, associated, authorized, endorsed by, or in any way officially connected with Space Exploration Technologies Inc (SpaceX), or any of its subsidiaries or its affiliates. The names SpaceX as well as related names, marks, emblems and images are registered trademarks of their respective owners.

Base URL

The most current version of the API is v3, with the following base URL `https://api.spacexdata.com/v3`

API Status

See the status page for details

Authentication

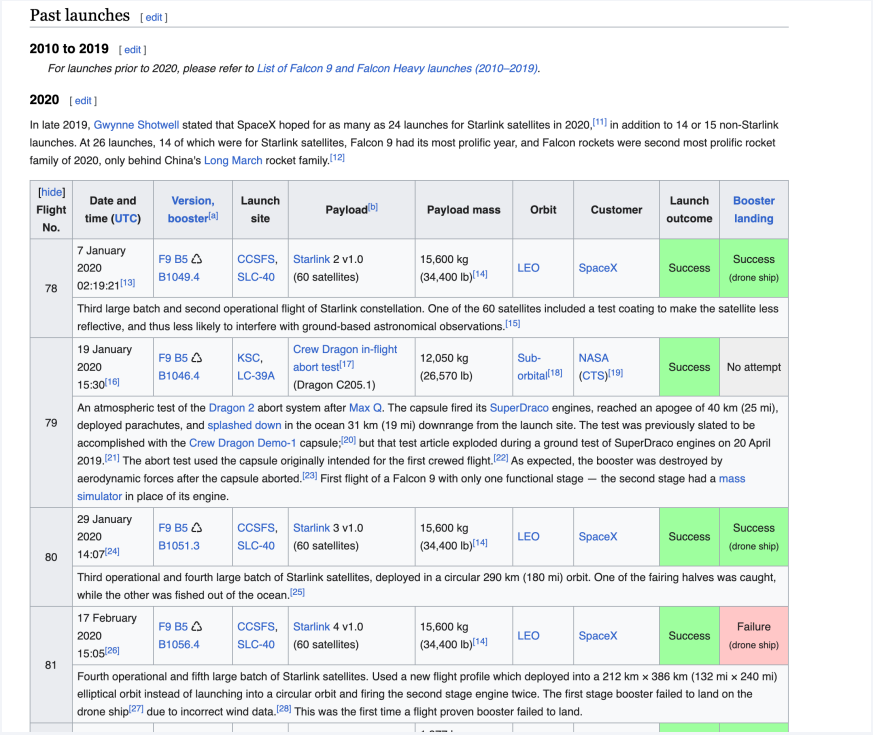
No authentication is required to use this public API

JSON Field Masking

Smaller JSON payloads can be generated through the use of the `filter` querystring. When using this querystring, all fields not included in the query will be omitted from the response.

For example, on the launches endpoint, you could include `filter=flight_number` to only return the flight number of every launch. Nested JSON fields can be expressed using a forward slash for each nested level. Ex: `filter=rocket/second_stage/payloads` to only return the payload objects from each launch. Multiple filters can be listed using a comma separator Ex: `filter=rocket/second_stage/payloads,flight_number,mission_name`. To select multiple properties within an object, use like so: `filter=rocket/second_stage/payloads/(payload_id,norad_id)`.

More information on the syntax can be found on the json-mask github page.



Past launches [edit]

2010 to 2019 [edit]

For launches prior to 2020, please refer to List of Falcon 9 and Falcon Heavy launches (2010–2019).

2020 [edit]

In late 2019, Gwynne Shotwell stated that SpaceX hoped for as many as 24 launches for Starlink satellites in 2020,[11] in addition to 14 or 15 non-Starlink launches. At 26 launches, 14 of which were for Starlink satellites, Falcon 9 had its most prolific year, and Falcon rockets were second most prolific rocket family of 2020, only behind China's Long March rocket family.[12]

[hide] Flight No.	Date and time (UTC)	Version, booster[a]	Launch site	Payload[b]	Payload mass	Orbit	Customer	Launch outcome	Booster landing
78	7 January 2020 02:19:21[13]	F9 B5 ♺ B1049.4	CCSFS, SLC-40	Starlink 2 v1.0 (60 satellites)	15,600 kg (34,400 lb)[14]	LEO	SpaceX	Success	Success (drone ship)
	Third large batch and second operational flight of Starlink constellation. One of the 60 satellites included a test coating to make the satellite less reflective, and thus less likely to interfere with ground-based astronomical observations.[15]								
79	19 January 2020 15:30[16]	F9 B5 ♺ B1046.4	KSC, LC-39A	Crew Dragon in-flight abort test[17] (Dragon C205.1)	12,050 kg (26,570 lb)	Sub-orbital[18]	NASA (CTS)[19]	Success	No attempt
	An atmospheric test of the Dragon 2 abort system after Max Q. The capsule fired its SuperDraco engines, reached an apogee of 40 km (25 mi), deployed parachutes, and splashed down in the ocean 31 km (19 mi) downrange from the launch site. The test was previously slated to be accomplished with the Crew Dragon Demo-1 capsule;[20] but that test article exploded during a ground test of SuperDraco engines on 20 April 2019.[21] The abort test used the capsule originally intended for the first crewed flight.[22] As expected, the booster was destroyed by aerodynamic forces after the capsule aborted.[23] First flight of a Falcon 9 with only one functional stage — the second stage had a mass simulator in place of its engine.								
80	29 January 2020 14:07[24]	F9 B5 ♺ B1051.3	CCSFS, SLC-40	Starlink 3 v1.0 (60 satellites)	15,600 kg (34,400 lb)[14]	LEO	SpaceX	Success	Success (drone ship)
	Third operational and fourth large batch of Starlink satellites, deployed in a circular 290 km (180 mi) orbit. One of the fairing halves was caught, while the other was fished out of the ocean.[25]								
81	17 February 2020 15:05[26]	F9 B5 ♺ B1056.4	CCSFS, SLC-40	Starlink 4 v1.0 (60 satellites)	15,600 kg (34,400 lb)[14]	LEO	SpaceX	Success	Failure (drone ship)
	Fourth operational and fifth large batch of Starlink satellites. Used a new flight profile which deployed into a 212 km × 386 km (132 mi × 240 mi) elliptical orbit instead of launching into a circular orbit and firing the second stage engine twice. The first stage booster failed to land on the drone ship[27] due to incorrect wind data.[28] This was the first time a flight proven booster failed to land.								

Data Collection – SpaceX API

Applying Open Source REST API for SpaceX:

1. Using GET request to parse the SpaceX launch data,
2. Extracting Data for Falcon 9 Launches,
3. Dealing with Missing Values by replacing them with mean values

[9]:
```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS032
```

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

[10]:
```
response.status_code
```

[10]: 200

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

[15]:
```
# Use json_normalize meethod to convert the json result into a dataframe
data = pd.json_normalize(response.json())
```

Using the dataframe `data` print the first 5 rows

[14]:
```
# Get the head of the dataframe
data.head()
```

[14]:

	static_fire_date_utc	static_fire_date_unix	net	window	rocket	success

Link:

https://github.com/kamalakbari7/Data-Science-Capstone-Project-IBM/blob/main/Notebooks/jupyter-labs-spacex-data-collection-api.ipynb

Data Collection - Scraping

Data collection with Web Scraping:

1. Requesting the Falcon9 Launch Wikipedia page from its URL
2. Extracting all column/variable names from the HTML table header
3. Creating a data frame by parsing the launch HTML tables

Link:

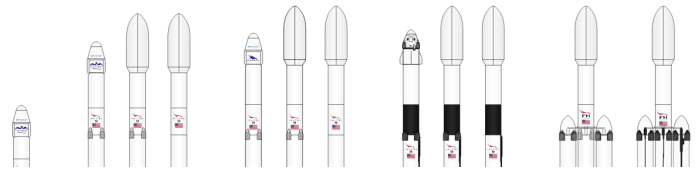
https://github.com/kamalakbari7/Data-Science-Capstone-Project-IBM/blob/main/Notebooks/jupyter-labs-webscraping.ipynb

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



[3]:
```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_la
```

Next, request the HTML page from the above URL and get a `response` object

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.

[4]:
```
# use requests.get() method with the provided static_url
# assign the response to a object
req = requests.get(static_url)
```

Create a `BeautifulSoup` object from the HTML `response`

[5]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(req.content,'html.parser')
```

Print the page title to verify if the `BeautifulSoup` object was created properly

[6]:
```
# Use soup.title attribute
soup.select('title')
```

[6]:
```
[<title>List of Falcon 9 and Falcon Heavy launches – Wikipedia</title>]
```

TASK 2: Extract all column/variable names from the HTML table header

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

Data Wrangling

- Exploratory Data Analysis (EDA) and determining the labels for training supervised models:
  1. Calculate the number of launches on each site
  2. Calculate the number and occurrence of each orbit
  3. Calculate the number and occurrence of mission outcome of the orbits
  4. Create a landing outcome label from the Outcome column

* Training Labels with 1 means the booster successfully landed, 0 means it was unsuccessful

Link:

https://github.com/kamalakbari7/Data-Science-Capstone-Project-IBM/blob/main/Notebooks/labs-jupyter-spacex-Data%20wrangling.ipynb

[28]:
```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
landing_class = []

for outcome in df['Outcome']:
    if outcome in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
```

This variable will represent the classification variable that represents the outcome of each launch. If the value is zero, the first stage did not land successfully; one means the first stage landed Successfully

[29]:
```
df['Class']=landing_class
df[['Class']].head(8)
```

[29]:

	Class
0	0
1	0
2	0
3	0
4	0
5	0
6	1
7	1

EDA with Data Visualization

Data Visualization of:

- Relationship between Flight Number and Launch Site
- Relationship between Payload and Launch Site
- Relationship between the success rate of each Orbit type
- Relationship between Flight Number and Orbit type
- Relationship between Payload and Orbit type
- The trend of yearly success rate

EDA with SQL

- Applying the sqlalchemy python package to connect to SQLite database
- Running SQL codes to show and list information about:
  - The names of the unique launch sites in the space mission
  - 5 records where launch sites begin with the string 'CCA'
  - The total payload mass carried by boosters launched by NASA (CRS)
  - Average payload mass carried by booster version F9 v1.1
  - The first successful landing outcome in the ground pad was achieved.
  - The boosters which have success in drone ships and have payload mass greater than 4000 but less than 6000
  - The total number of successful and failed mission outcomes
  - The names of the booster_versions which have carried the maximum payload mass. Use a subquery
  - The records, which will display the month names, failure landing_outcomes in drone ship, booster versions, and launch_site for the months in the year 2015

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

```
[8]: %sql SELECT DISTINCT "LAUNCH_SITE" FROM SPACEXTBL;
 * sqlite:///my_data1.db
Done.
```

[8]:

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

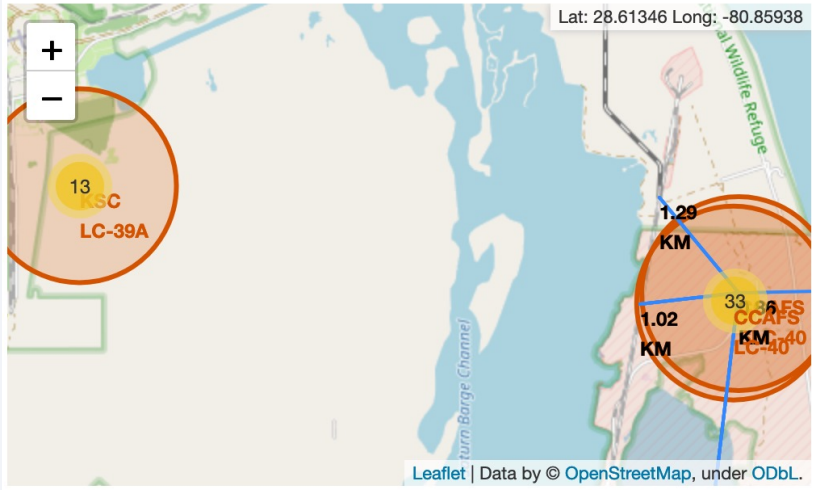
None

Link:
https://github.com/kamalakbari7/Data-Science-Capstone-Project-IBM/blob/main/Notebooks/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- Applying the Folium Python package to map and visualize:
  - All launch sites on a map
  - The successful/failed launches for each site on the map
  - Distances between a launch site to its proximities, such as:
    - Cities
    - Railways
    - Highways
    - Coastlines

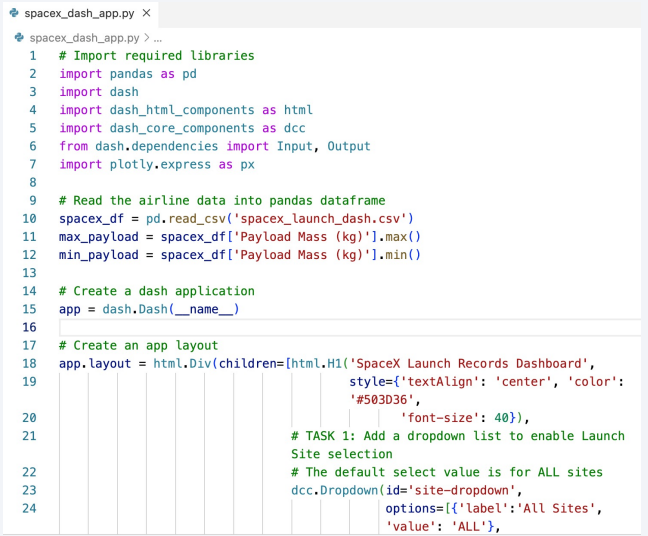
```
# Create a blue circle at NASA Johnson Space Center's coordinate v
circle = folium.Circle(nasa_coordinate, radius=1000, color='#d3540
# Create a blue circle at NASA Johnson Space Center's coordinate v
marker = folium.map.Marker(
    nasa_coordinate,
    # Create an icon as a text label
    icon=DivIcon(
        icon_size=(30,30),
        icon_anchor=(0,0),
        html='<div style="font-size: 12; color:#d35400;"><b>%s</b>
        )
    )
site_map.add_child(circle)
site_map.add_child(marker)
```



Link:
https://github.com/kamalakbari7/Data-Science-Capstone-Project-IBM/blob/main/Notebooks/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

- Using Plotly Dash application to perform interactive visual analytics on SpaceX launch data
- A Launch Site Drop-down Input Component to select different launch sites
- A Pie chart to show the percentage of missions from different sites and the success rate of missions in the selected site(s)
- A Range Slider to select payload to define the range of targeted payloads
- A Scatter plot to show the correlation between payload and mission outcomes for the selected site(s)


spacex_dash_app.py

```python
# Import required libraries
import pandas as pd
import dash
import dash_html_components as html
import dash_core_components as dcc
from dash.dependencies import Input, Output
import plotly.express as px

# Read the airline data into pandas dataframe
spacex_df = pd.read_csv('spacex_launch_dash.csv')
max_payload = spacex_df['Payload Mass (kg)'].max()
min_payload = spacex_df['Payload Mass (kg)'].min()

# Create a dash application
app = dash.Dash(__name__)

# Create an app layout
app.layout = html.Div(children=[html.H1('SpaceX Launch Records Dashboard',
                                        style={'textAlign': 'center', 'color':
                                        '#503D36',
                                               'font-size': 40}),
                                # TASK 1: Add a dropdown list to enable Launch
                                Site selection
                                # The default select value is for ALL sites
                                dcc.Dropdown(id='site-dropdown',
                                             options=[{'label':'All Sites',
                                             'value': 'ALL'},
```

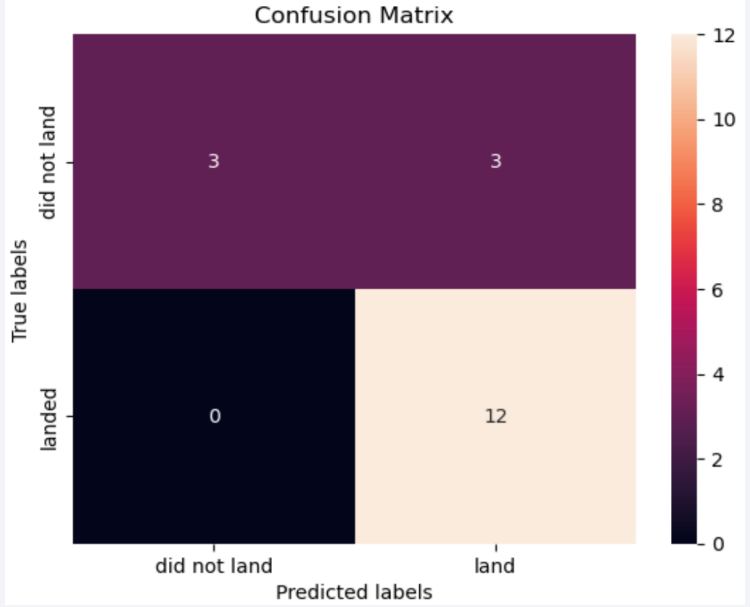
Link:
https://github.com/kamalakbari7/Data-Science-Capstone-Project-IBM/blob/main/Scripts/spacex_dash_app.py

Predictive Analysis (Classification)

- Read and standardize the required data
- Split the data to training and test datasets
- Apply different prediction methods to the training data
    - •Logistic Regression
    - •Support Vector Machines
    - •Decision Trees
    - •K Nearest Neighbors
- Applying cross-validated grid-search (GridSearchCV) to select the best hyperparameters based on training data
- Evaluate the accuracy of the models and evaluate the confusion matrix for models

```
logreg_cv = GridSearchCV(lr,parameters,cv=10)
logreg_cv.fit(X_train,Y_train)
```

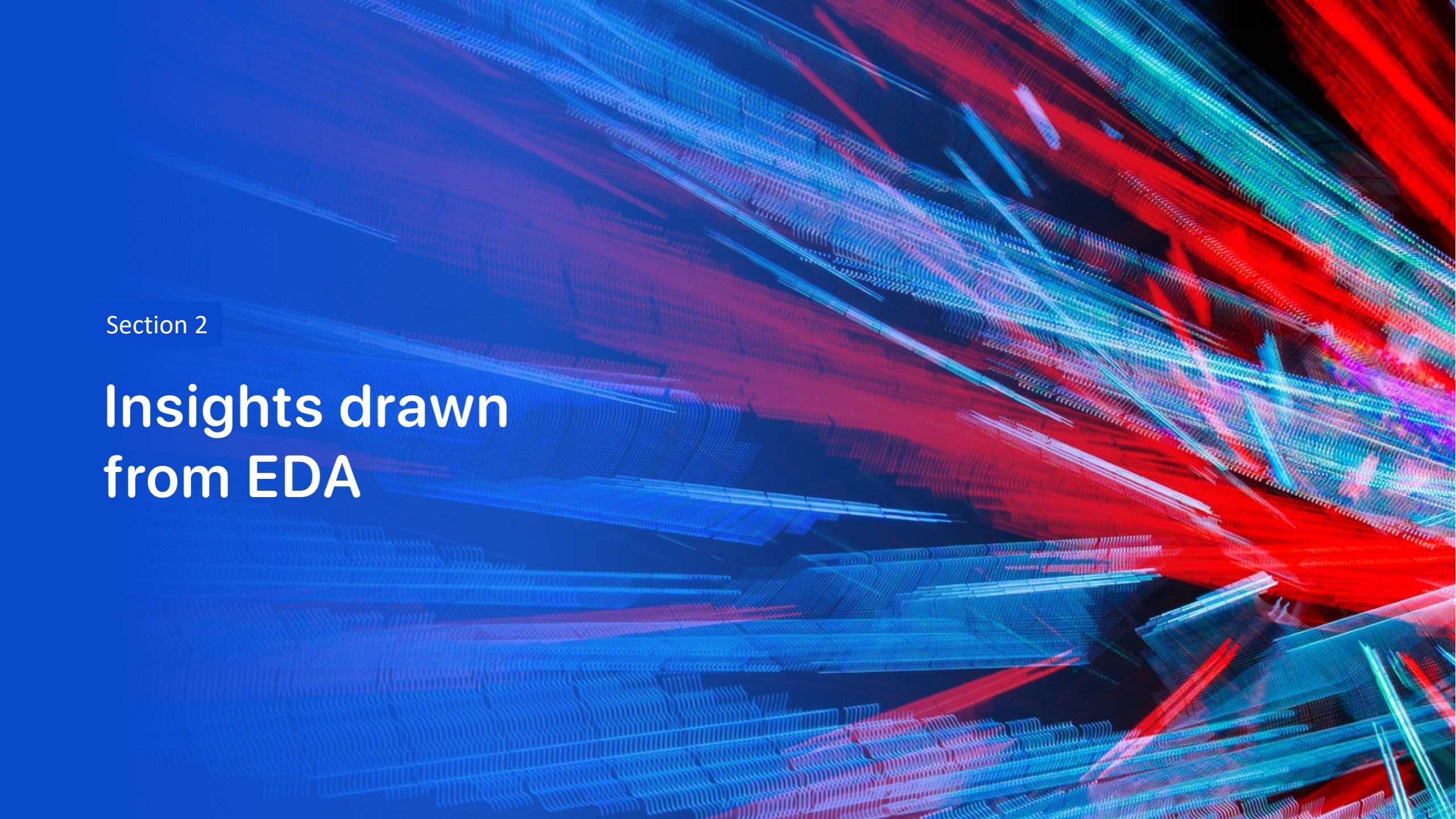
```
GridSearchCV
GridSearchCV(cv=10, estimator=LogisticRegression(),
             param_grid={'C': [0.01, 0.1, 1], 'penalty':
['l2'],
                         'solver': ['lbfgs']})
estimator: LogisticRegression
LogisticRegression()
LogisticRegression
LogisticRegression()
```



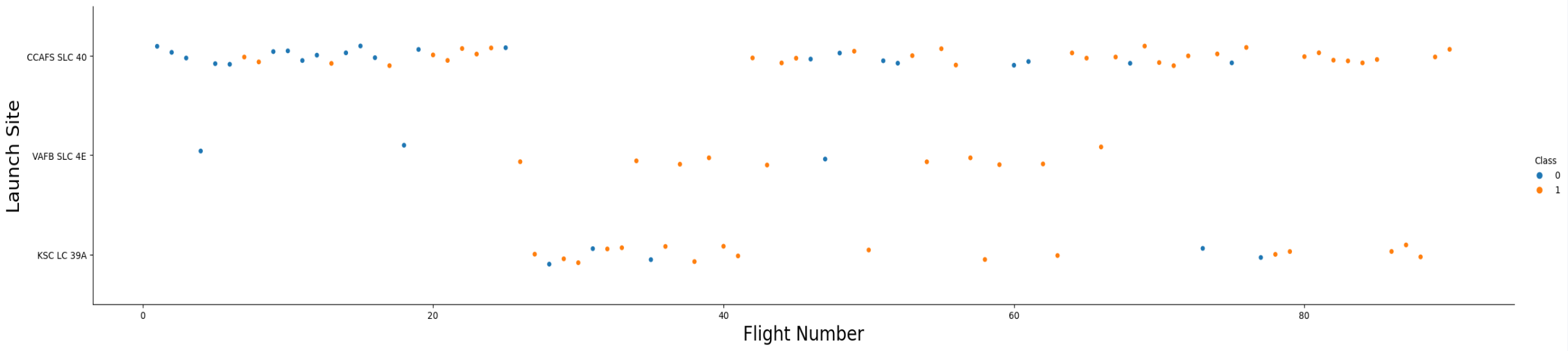
Link: https://github.com/kamalakbari7/Data-Science-Capstone-Project-IBM/blob/main/Notebooks/SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

Results

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

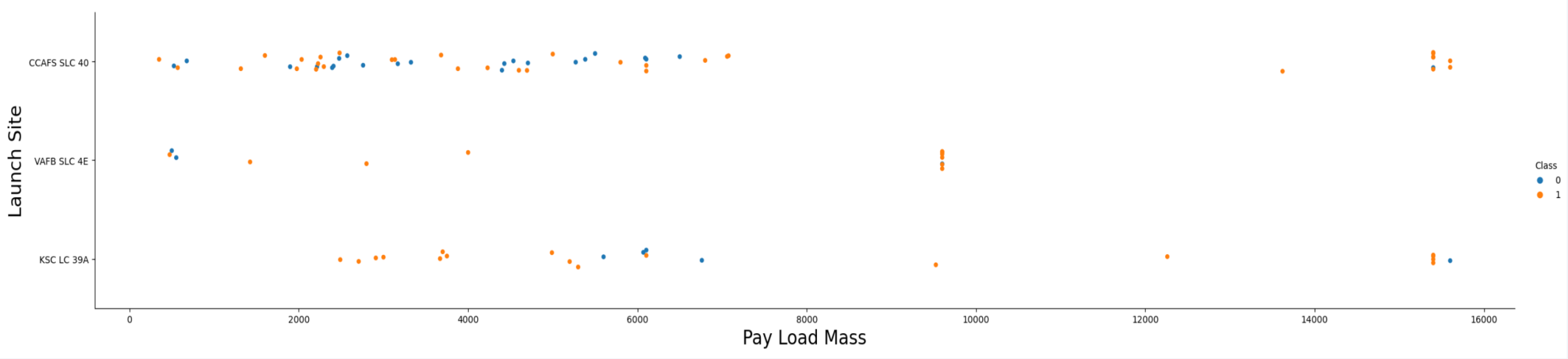

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site



- The graph shows an increase in success rate by time for all launch sites
- CCAFS SLC 40 has the lowest rate of success in missions

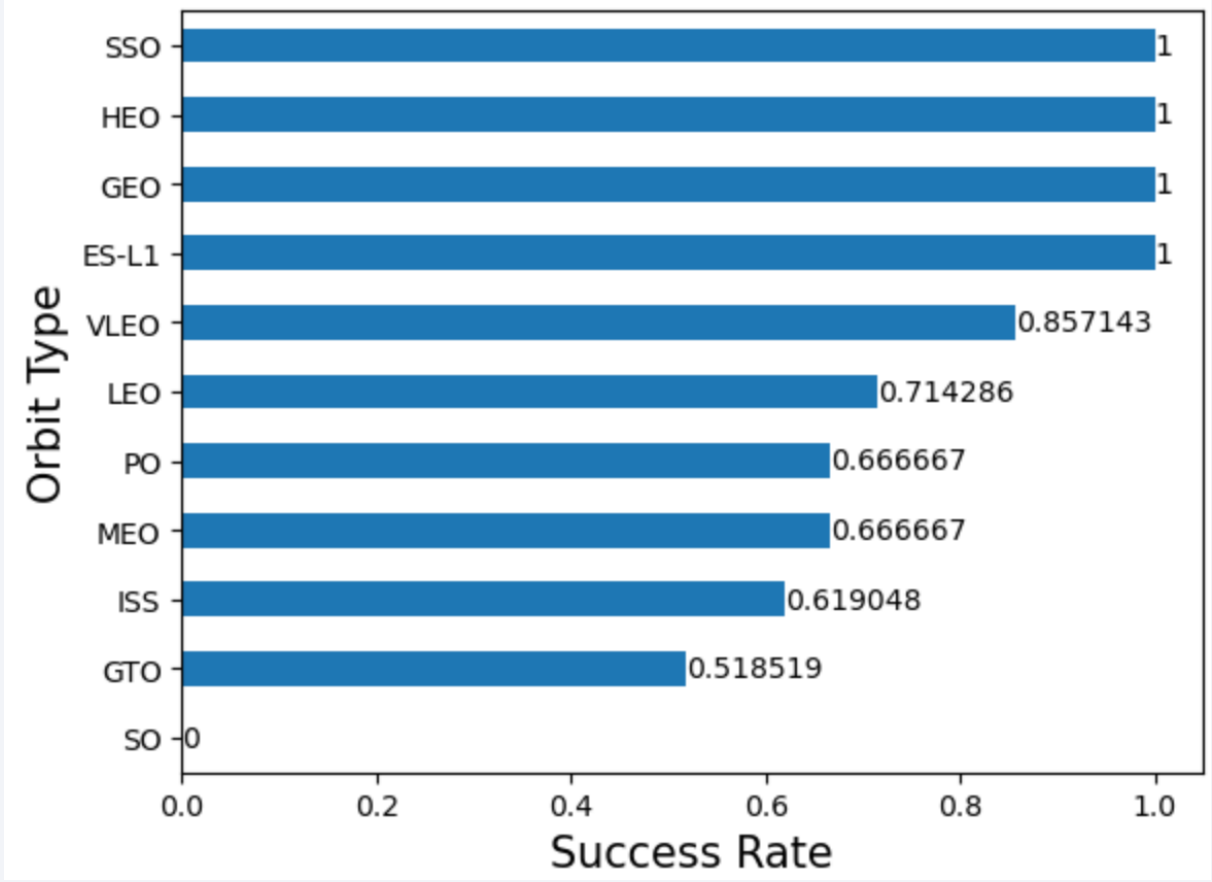
Payload vs. Launch Site



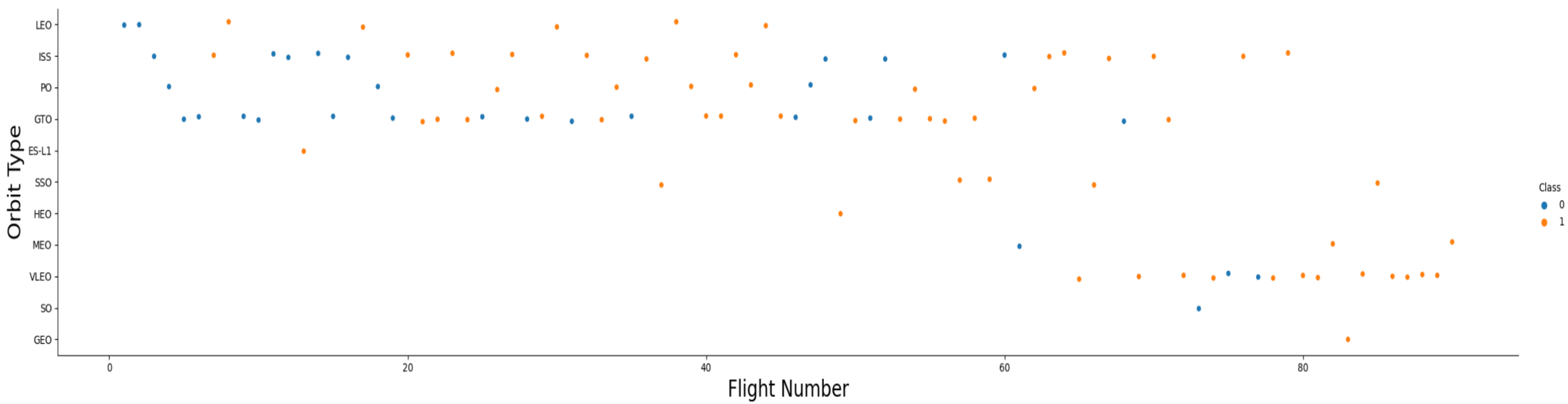
- There is a higher success rate for higher payloads
- KSC LC 39A and VAFB SLC 4E have had high success rates in lower payloads despite CCAFS SLC 40

Success Rate vs. Orbit Type

- There are four orbit types with the highest success rate
- SO has the lowest success rate, with 0%
- The number of launches is the key variable that should be considered for interpreting these results

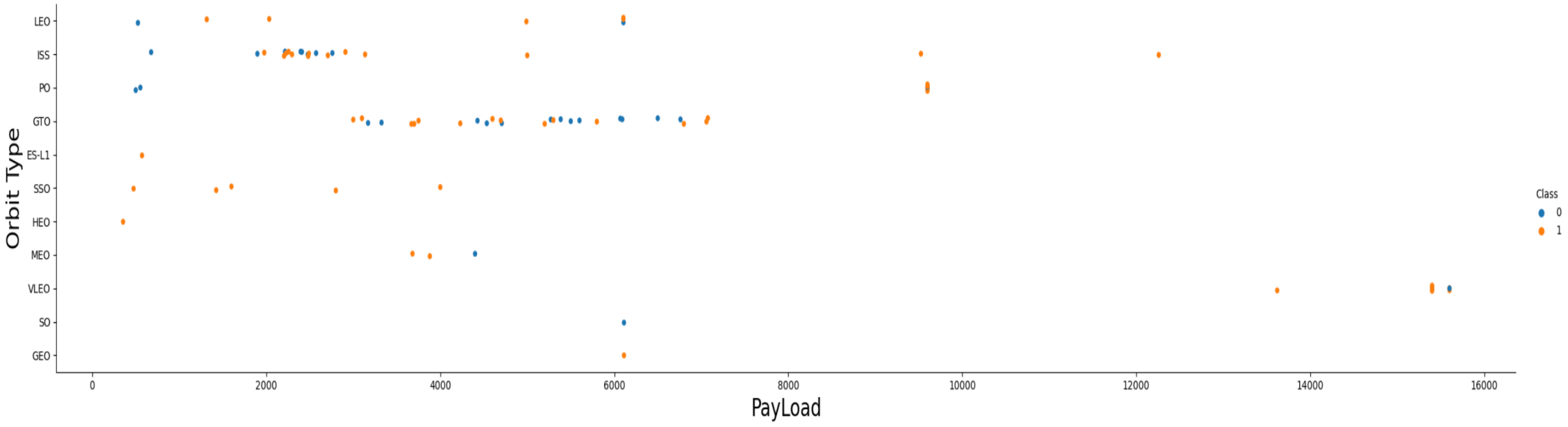


Flight Number vs. Orbit Type



- Based on the success rate and number of launches, SSO is the most successful mission for SpaceX
- The success rate increases with time
- GTO and ISS orbit types have the highest number of launches

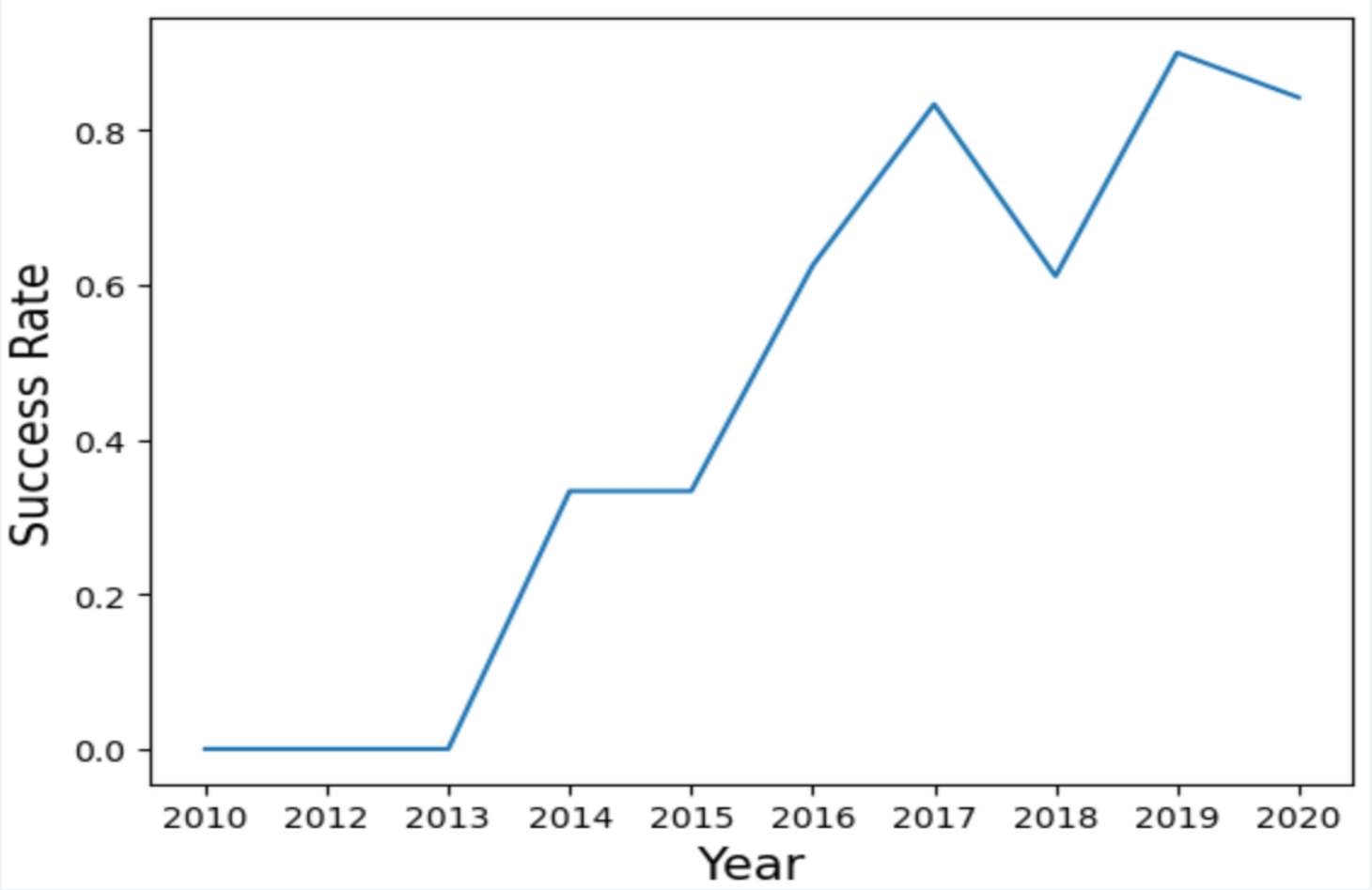
Payload vs. Orbit Type



- Most of the launches for orbit types have payload mass less than 8000
- SSO with the highest success rate has payloads less than 4000
- Higher payload masses have been more successful than lower masses

Launch Success Yearly Trend

- There is an increase in success rate after 2013
- A significant decrease in the success rate can be observable in 2018



All Launch Site Names

- The list of unique launch sites by applying the “distinct” statement in SQL

```
%sql SELECT DISTINCT "LAUNCH_SITE" FROM SPACEXTBL;
```

```
 * sqlite:///my_data1.db
Done.
```

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

Launch Site Names Begin with 'CCA'

- Selecting 5 records where launch sites begin with `CCA`
- We can get the requested results by applying LIKE function on `CCA%` and limit the results for 5 records

```
%sql SELECT * FROM SPACEXTBL WHERE "LAUNCH_SITE" LIKE 'CCA%' LIMIT 5;
```

```
 * sqlite:///my_data1.db
Done.
```

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
06/04/2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0.0	LEO	SpaceX	Success	Failure (parachute)
12/08/2010	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0.0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
22/05/2012	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525.0	LEO (ISS)	NASA (COTS)	Success	No attempt
10/08/2012	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500.0	LEO (ISS)	NASA (CRS)	Success	No attempt
03/01/2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677.0	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- The total payload carried by boosters from NASA is achievable by applying SUM function and WHERE clause

```
%sql SELECT SUM("PAYLOAD_MASS__KG_") FROM SPACEXTBL WHERE "CUSTOMER"="NASA (CRS)";
```

```
 * sqlite:///my_data1.db
Done.
```

SUM("PAYLOAD_MASS__KG_")
45596.0

Average Payload Mass by F9 v1.1

- The average payload mass carried by booster version F9 v1.1 can be calculated by applying AVG function

```
%%sql
SELECT AVG("PAYLOAD_MASS__KG_") FROM SPACEXTBL WHERE "Booster_Version" LIKE 'F9 v1.1%';
```

```
 * sqlite:///my_data1.db
Done.
```

AVG("PAYLOAD_MASS__KG_")
2534.6666666666665

First Successful Ground Landing Date

- The date of the first successful landing outcome on a ground pad can be achievable by applying the MIN function on the DATE column

```
%%sql
SELECT MIN("DATE") FROM SPACEXTBL WHERE "LANDING_OUTCOME" = 'Success (ground pad)';
```

```
 * sqlite:///my_data1.db
Done.
```

MIN("DATE")
01/08/2018

Successful Drone Ship Landing with Payload between 4000 and 6000

- The list of boosters which have successfully landed on drone ships and had payload mass greater than 4000 but less than 6000 can be explored by applying the DISTINCT statement and BETWEEN function

```
%%sql
SELECT DISTINCT("Booster_Version") FROM SPACEXTBL WHERE "Landing_Outcome" = 'Success (drone ship)' AND "PAYLOAD_MASS__KG_" BETWEEN 4000.0 AND 6000.0;
```

```
 * sqlite:///my_data1.db
Done.
```

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

Total Number of Successful and Failure Mission Outcomes

- The total number of successful and failed mission outcomes can be calculated by applying the COUNT function and GROUP BY statement

```
%%sql
SELECT DISTINCT("Mission_Outcome"),COUNT("Mission_Outcome") AS "TOTAL_NUMBER" FROM SPACEXTBL
GROUP BY "Mission_Outcome";
```

```
 * sqlite:///my_data1.db
Done.
```

Mission_Outcome	TOTAL_NUMBER
None	0
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Boosters Carried Maximum Payload

- A list of the booster which has carried the maximum payload mass can be achievable by applying a subquery with the MAX function

```
%%sql
SELECT DISTINCT("Booster_Version")
FROM SPACEXTBL
WHERE "PAYLOAD_MASS__KG_"= (SELECT MAX("PAYLOAD_MASS__KG_") FROM SPACEXTBL);
```

```
 * sqlite:///my_data1.db
Done.
```

Booster_Version
F9 B5 B1048.4
F9 B5 B1049.4
F9 B5 B1051.3
F9 B5 B1056.4
F9 B5 B1048.5
F9 B5 B1051.4
F9 B5 B1049.5
F9 B5 B1060.2
F9 B5 B1058.3
F9 B5 B1051.6
F9 B5 B1060.3
F9 B5 B1049.7

2015 Launch Records

- The list the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015 was done by applying LIKE function

```
%%sql
SELECT SUBSTR("DATE",4,2) AS MONTH,"LANDING_OUTCOME", "Booster_Version","Launch_Site"
FROM SPACEXTBL
WHERE SUBSTR("DATE",7,4)='2015' AND "Landing_Outcome" ="Failure (drone ship)";
```

```
 * sqlite:///my_data1.db
Done.
```

MONTH	Landing_Outcome	Booster_Version	Launch_Site
10	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

- Rank 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.
- We need to use COUNT, WHERE, BETWEEN, GROUP BY, and ORDER BY

```
%%sql
SELECT "Landing_Outcome", COUNT("Landing_Outcome") AS "TOT_NUMBER"
FROM SPACEXTBL
WHERE "DATE" BETWEEN '04/06/2010' AND '20/03/2017'
GROUP BY "Landing_Outcome"
ORDER BY "TOT_NUMBER" DESC;
```

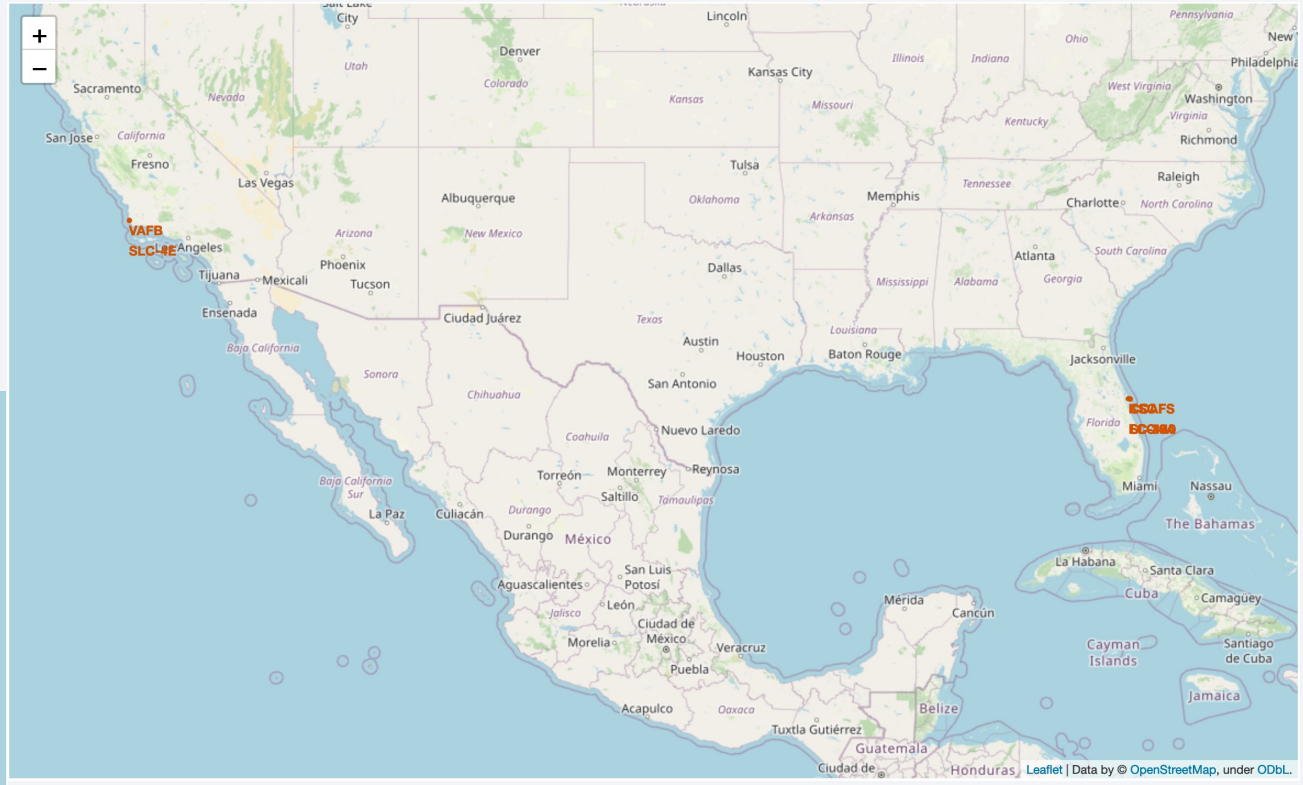
```
 * sqlite:///my_data1.db
Done.
```

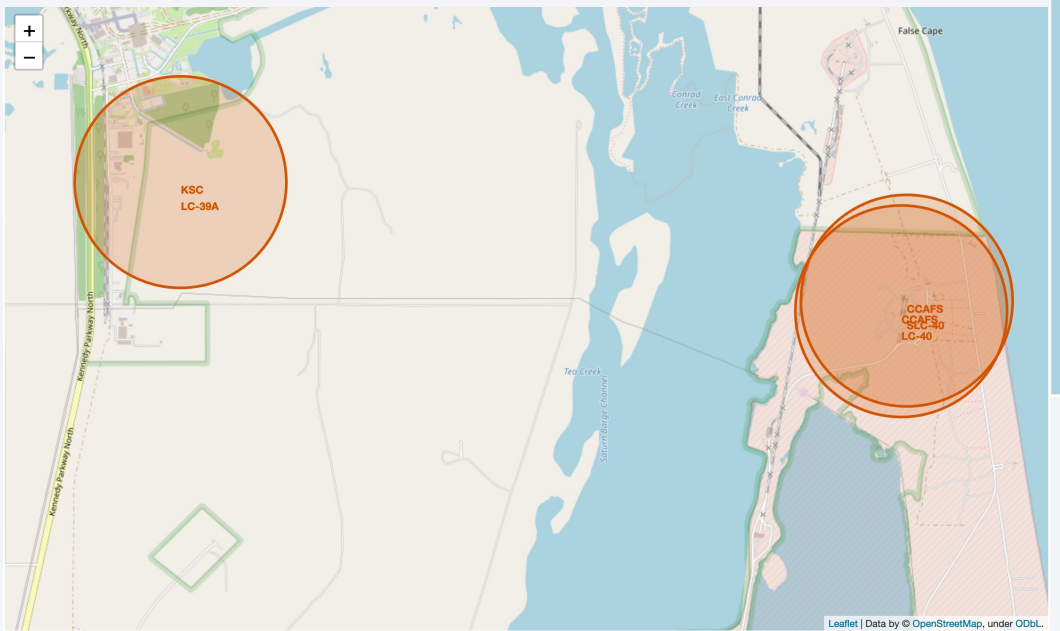
Landing_Outcome	TOT_NUMBER
Success	20
No attempt	9
Success (drone ship)	8
Success (ground pad)	7
Failure (drone ship)	3
Failure	3
Failure (parachute)	2
Controlled (ocean)	2
No attempt	1


Section 3
Launch Sites
Proximities Analysis

Launch Sites Locations

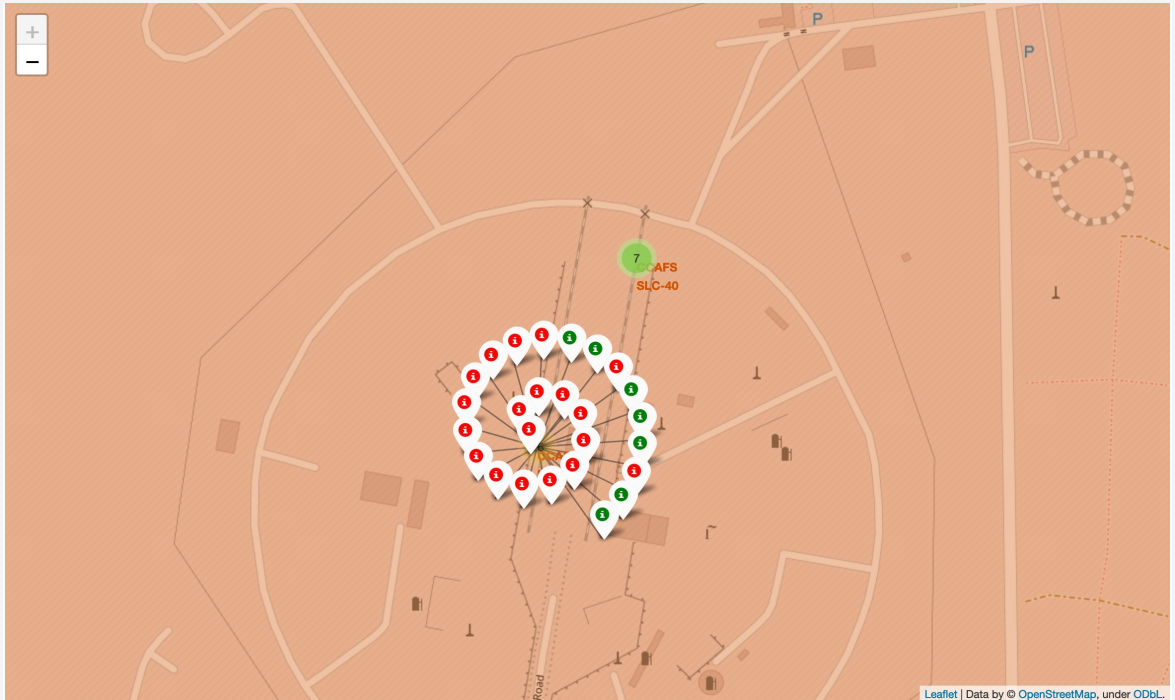
- Launch sites are located in the vicinity of the oceans
- There is only one site in the west, and three sites are in the east





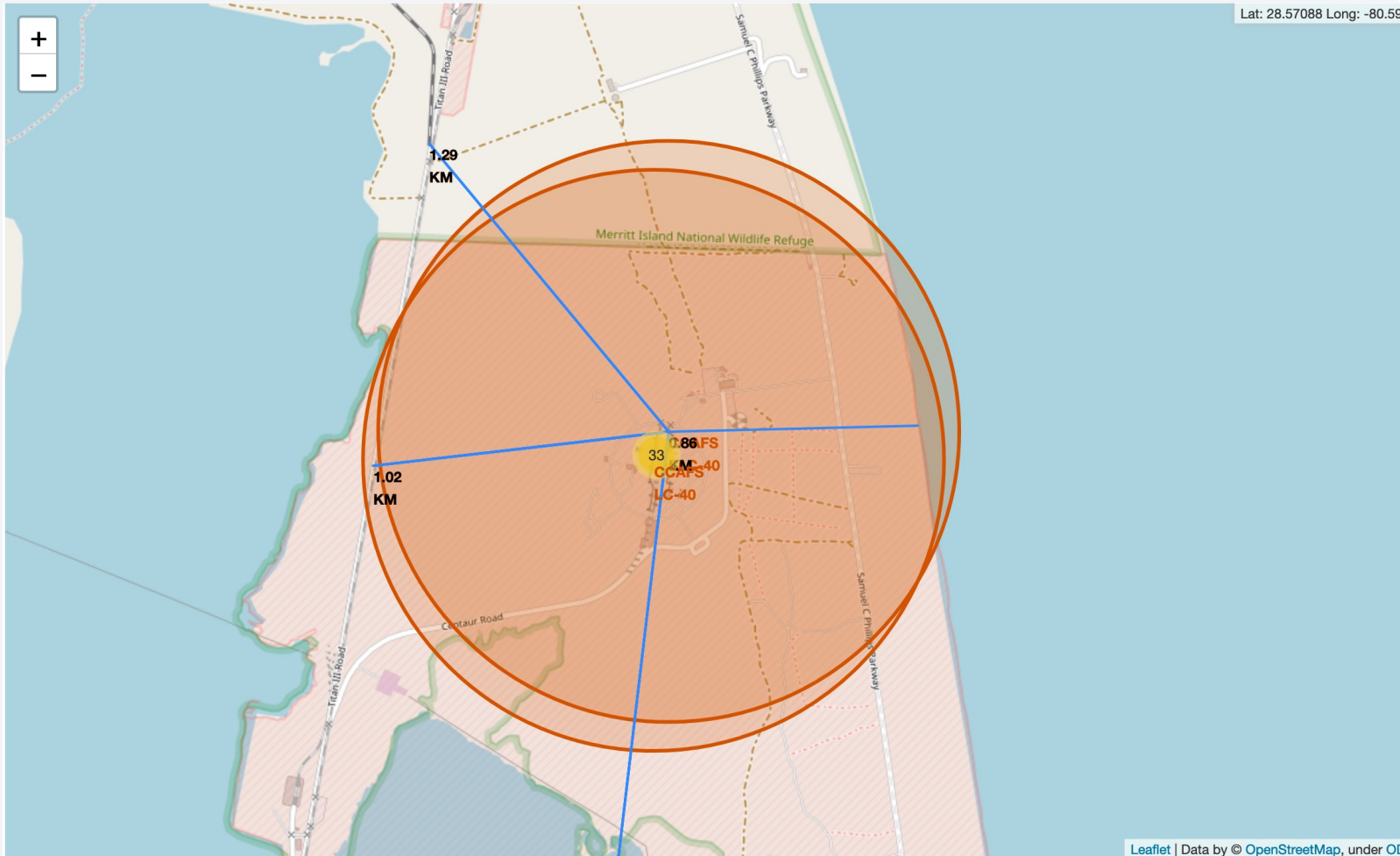
Status of Launches for Sites

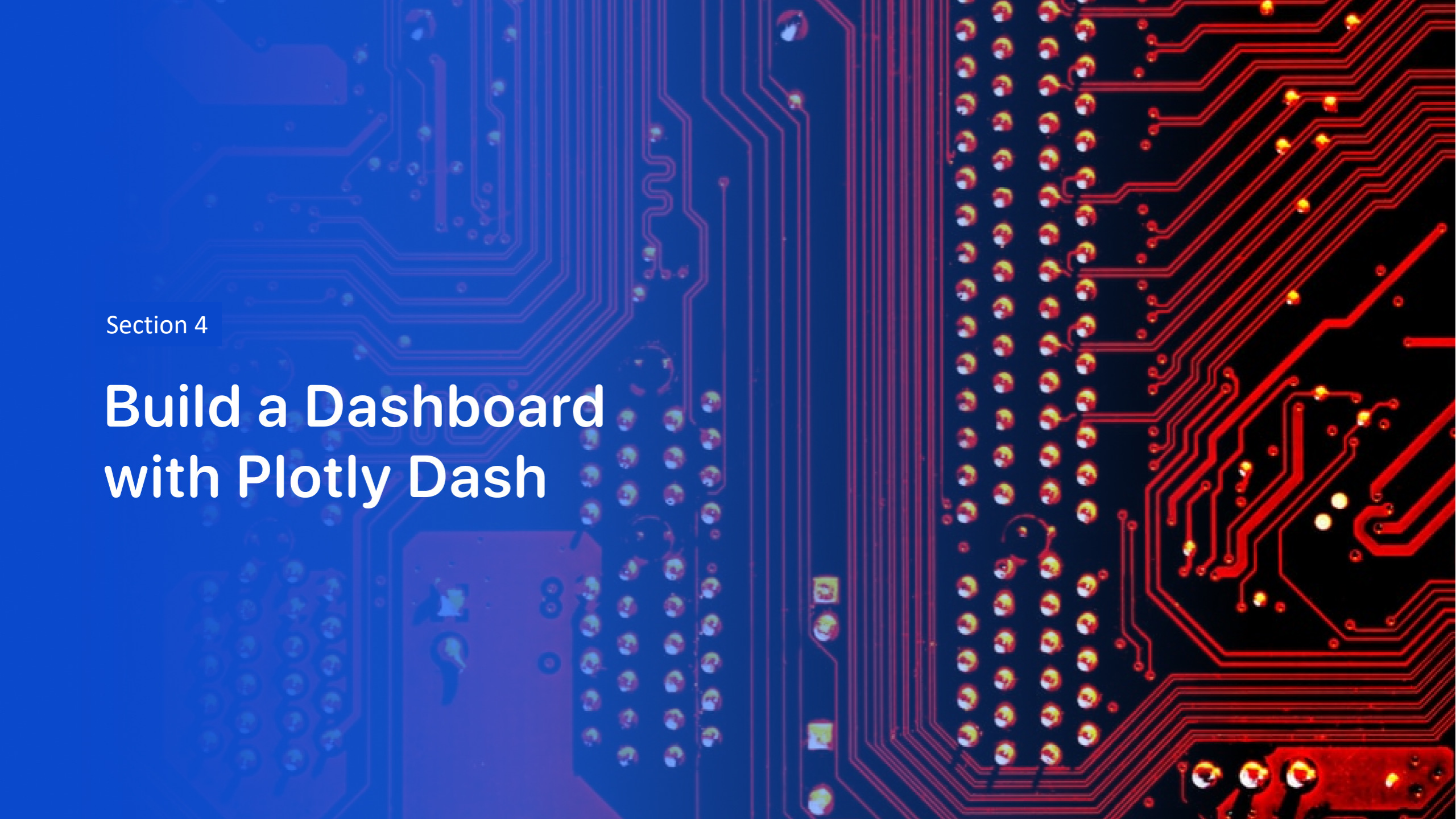
- In the right map, we have clusters with yellow colour to show the missions' locations
- In each cluster, we have successful missions with green colour and failed missions with red colour (left map)



Proximities of a Launch Site

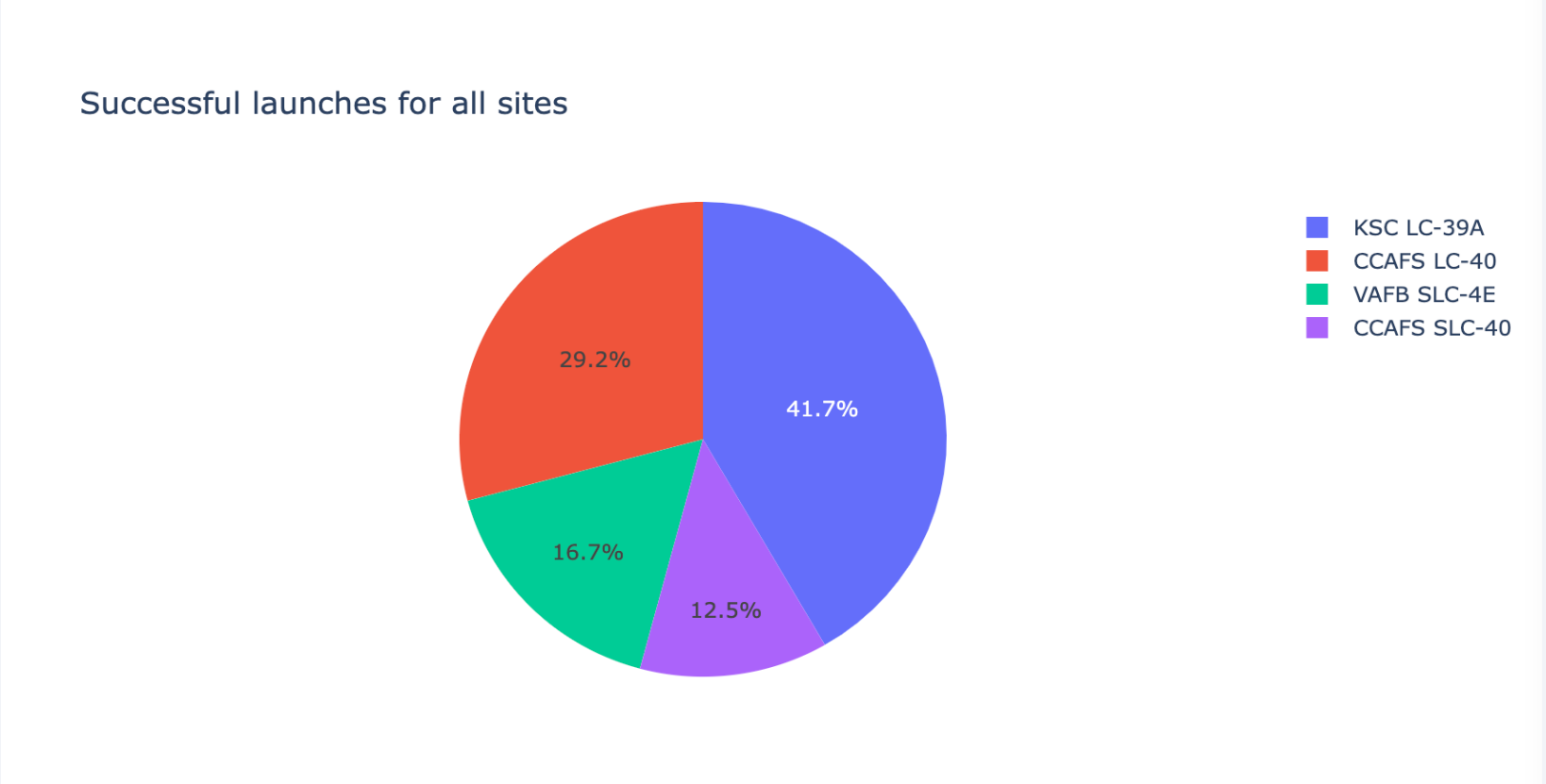
- There is a 51 km distance between this site and the nearest city
- The nearest railways, roads, and highways are located 1.3 km from the site location
- These distances from networks show the importance of accessibility to supply, and the distance from the city shows the importance of safety




Section 4
Build a Dashboard with Plotly Dash

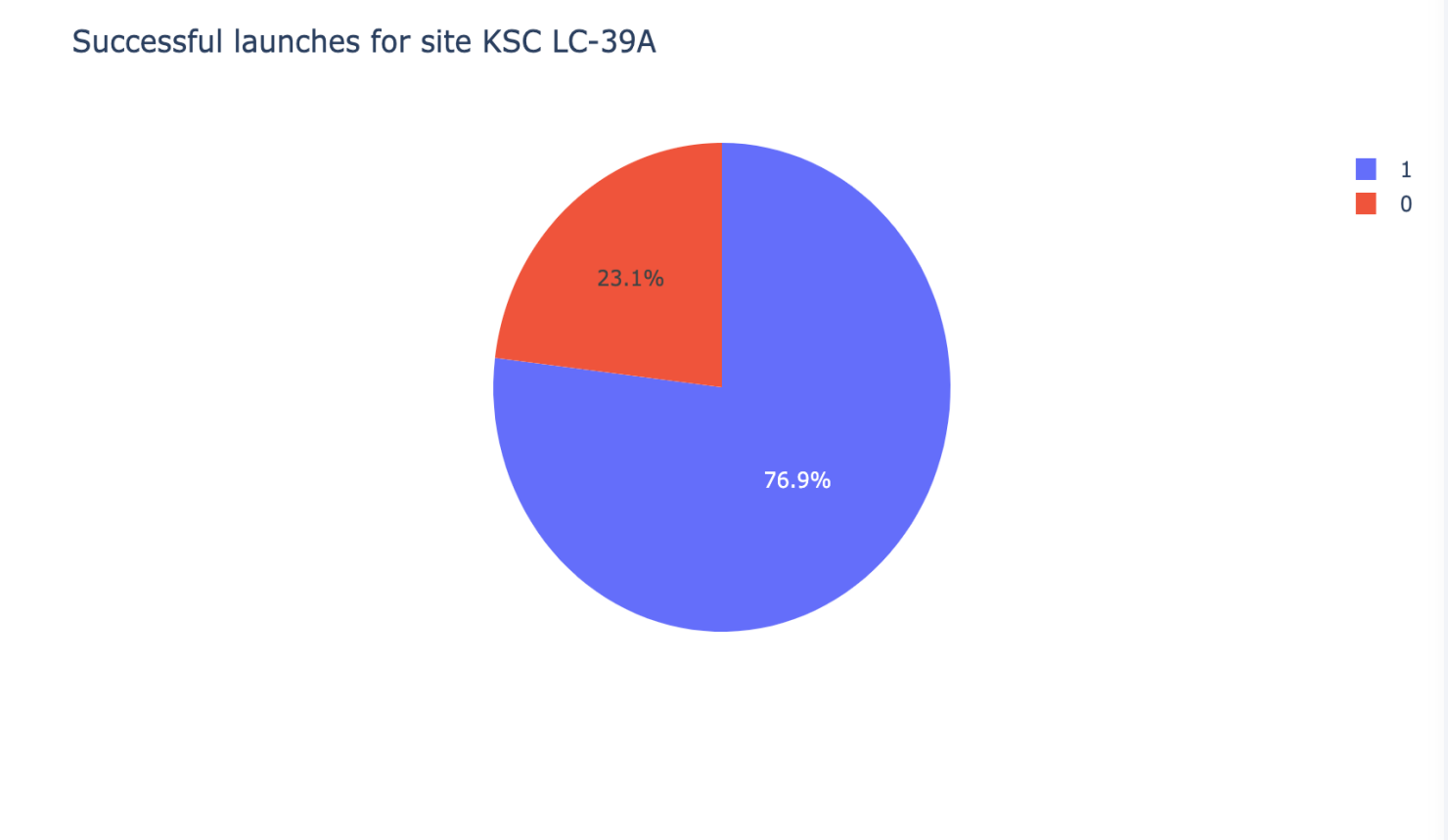
Successful Launches for All Sites

- KCS LC-39A and CCAFS SLC-40 have the highest and lowest number of successful launches respectively



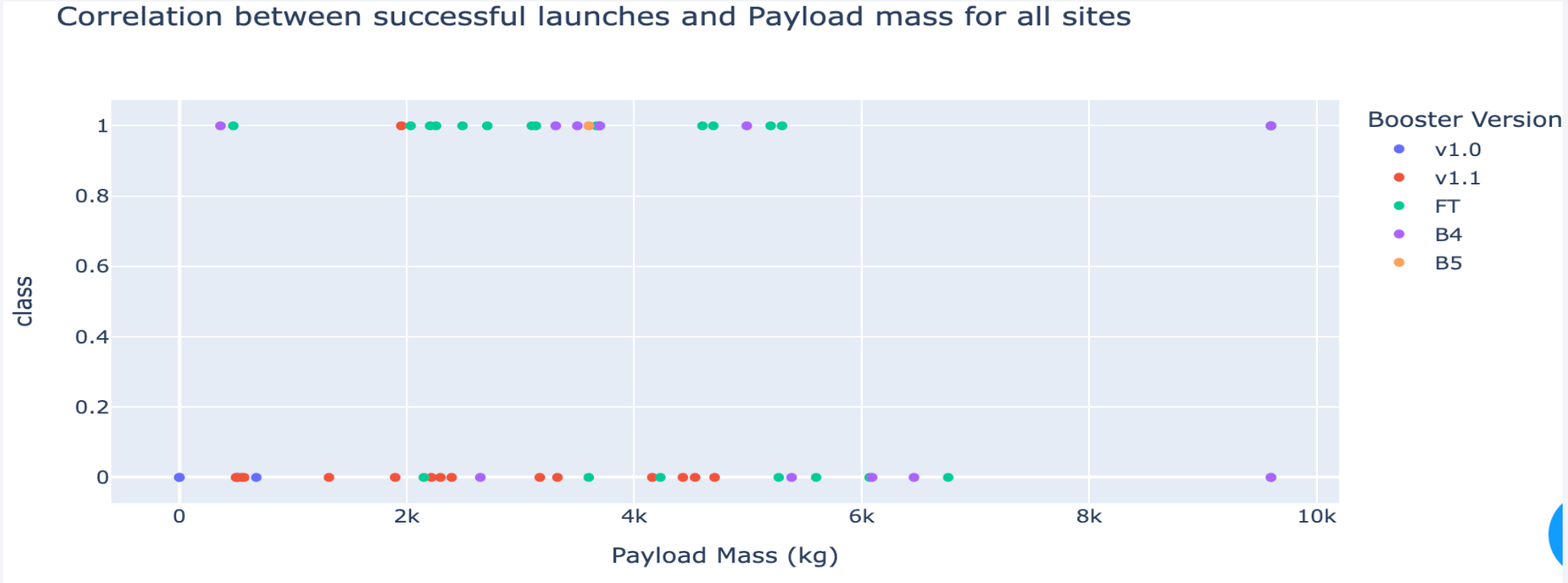
Success and Failure for KSC LC-39A

- KSC LC-39A site with the highest success rate

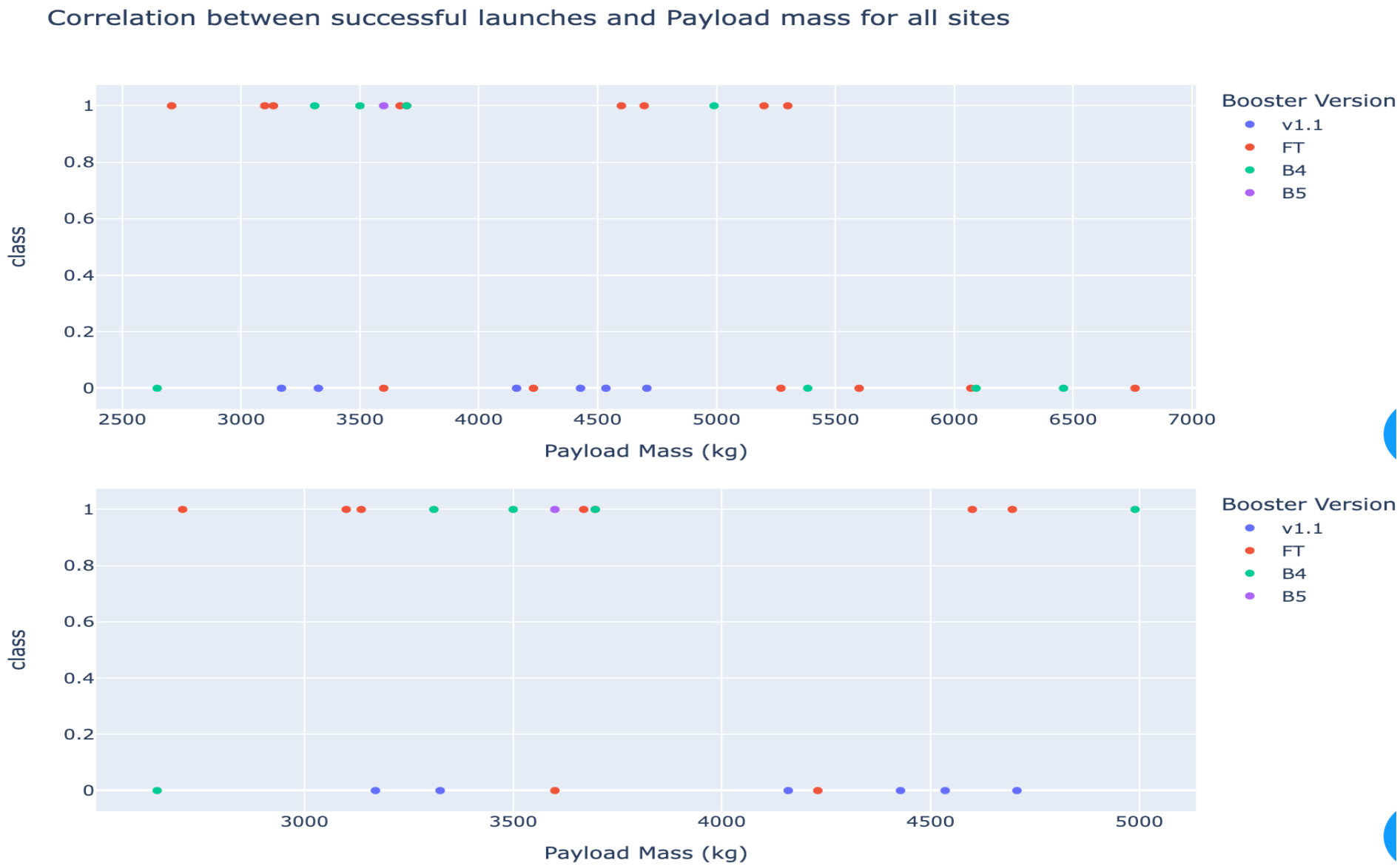


Payload Mass vs. Launch Outcome

- Most of the successful launches are for missions with payloads between 2000 kg to 6000 kg
- V1.1 and FT have the lowest and highest success rate, respectively



Payload Mass vs. Launch Outcome

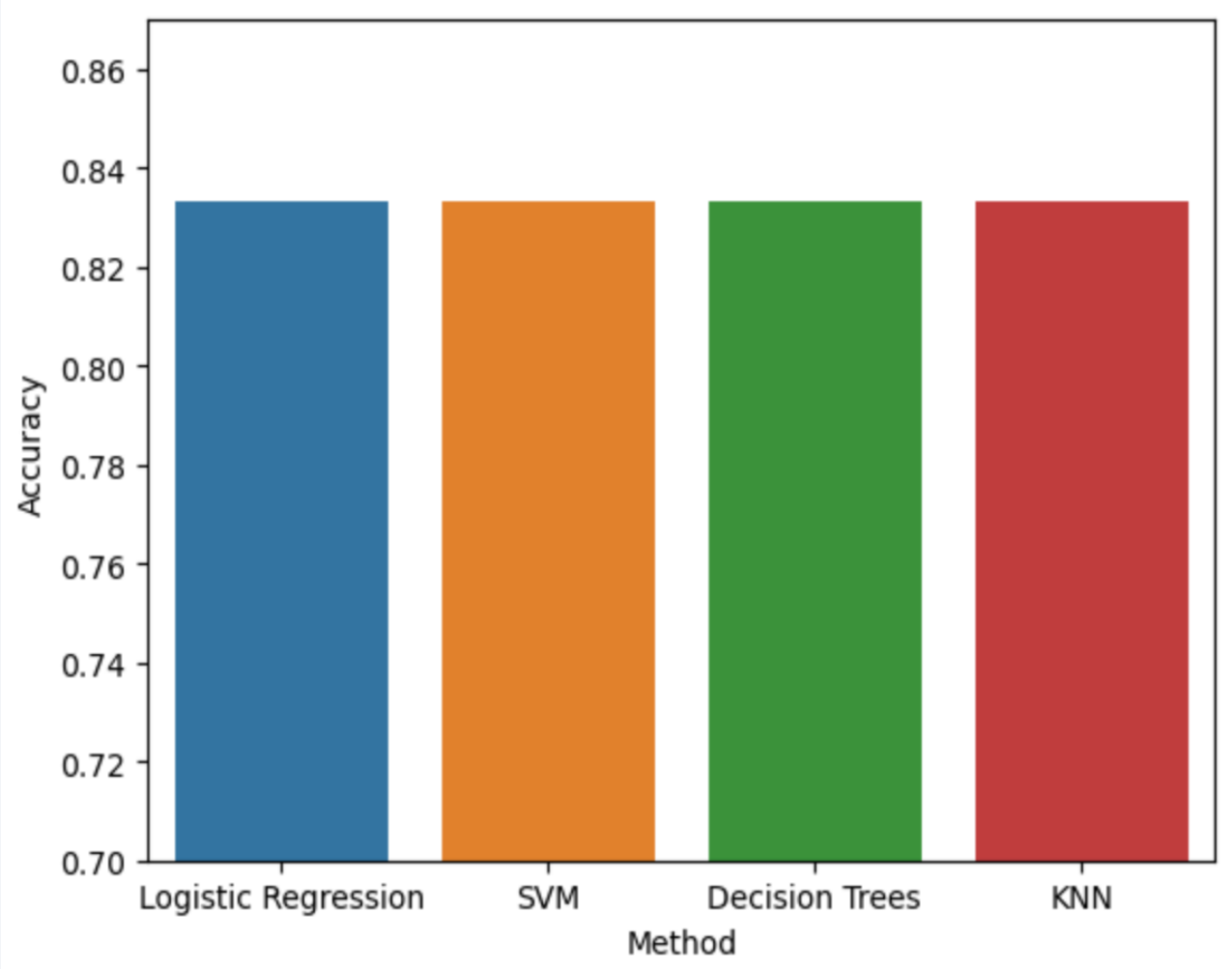


Section 5

Predictive Analysis (Classification)

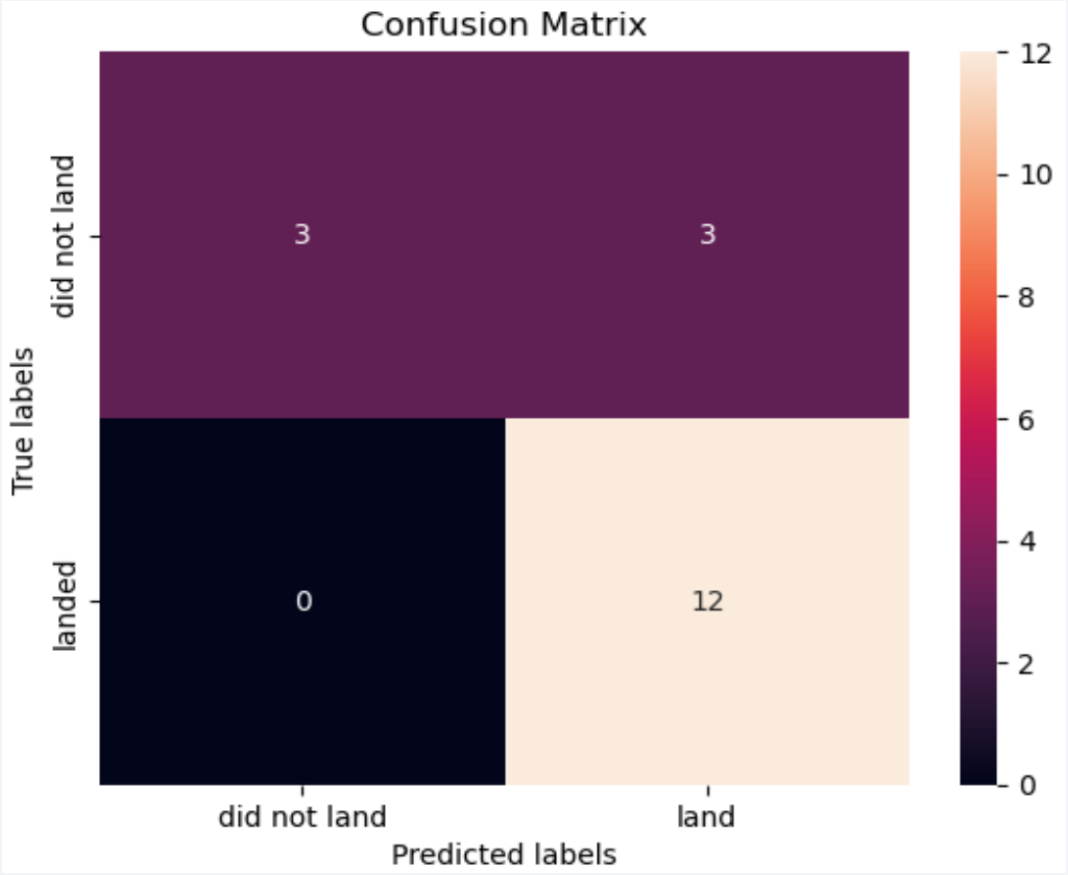
Classification Accuracy

- All implemented predictive models have similar accuracy



Confusion Matrix

- All models have similar confusion matrix



Conclusions

- With the results of predictive models as SpaceY company, we can predict the success rate of our missions up to 83 %
- As SpaceY, we can use results to estimate the cost of our missions
- With exploratory analysis, as SpaceY company, we know the features of site launches and can design our infrastructures better
- These results were based on a limited dataset, and we may increase the accuracy of the models by applying new supplementary data
- These exploratory and analytical studies on SpaceX missions can help SpaceY to define the main goals and find the right directions for investment

Appendix

- https://github.com/kamalakbari7/Data-Science-Capstone-Project-IBM/tree/main

Thank you!

Kamalakbari77@gmail.com