IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Mahmoud Almohammad
09 Mar 2024

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection through API
  - Data Collection with Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis with SQL
  - Exploratory Data Analysis with Data Visualization
  - Interactive Visual Analytics with Folium
  - Machine Learning Prediction
- Summary of all results
  - Exploratory Data Analysis result
  - Interactive analytics in screenshots
  - Predictive Analytics result from Machine Learning Lab

Introduction

Project: Predicting Rocket First-Stage Landing Outcomes with Machine Learning

This project aims to develop a machine learning model to forecast the success of a rocket's first-stage landing. By analyzing various factors that influence landing, the model can provide valuable insights for launch providers.

Key Challenges:

Identifying all relevant factors affecting landing success, such as weather conditions, launch trajectory, rocket health, and stage separation conditions.

Understanding the relationships between these variables and their impact on the landing outcome.

Determining the optimal conditions that maximize the probability of a successful landing.

Introduction

Potential Benefits:

Improved launch success rates through informed decision-making.

More efficient mission planning and resource allocation.

Development of competitive pricing strategies for launch services.

This project focuses on building a robust model to predict first-stage landing outcomes. The information gained can be instrumental for launch providers in optimizing their operations and achieving greater success rates.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using SpaceX REST API and web scrapping from Wikipedia
- Perform data wrangling
  - Data was processed using one-hot encoding 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
  - How to build, tune, evaluate classification models

Data Collection

In this project, data was collected from Wikipedia using two methods: REST API and web scraping.

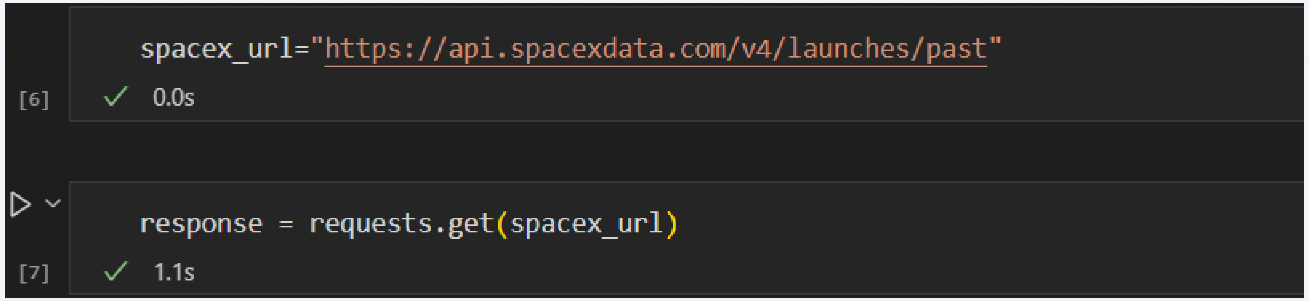
REST API: We used a GET request to retrieve data in JSON format. The JSON response was then converted into a pandas DataFrame using the json_normalize function. Following this, the data was cleaned, missing values were identified, and filled as necessary.

Web Scraping:

BeautifulSoup was employed to extract launch records presented as HTML tables. The tables were parsed and converted into a pandas DataFrame for further analysis.

Data Collection - SpaceX API

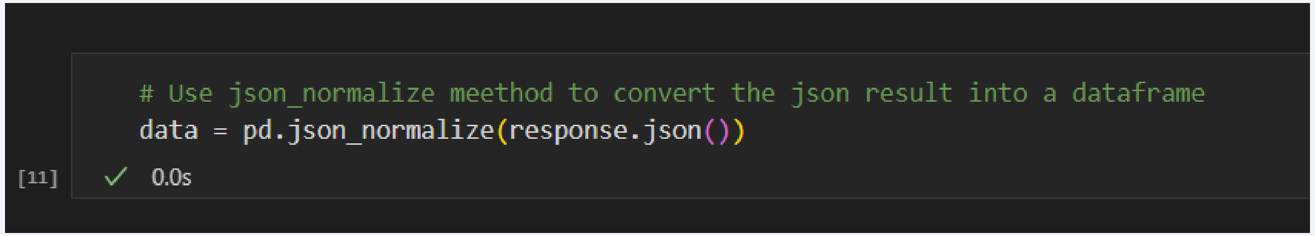
Get request for rocket launch data using API



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

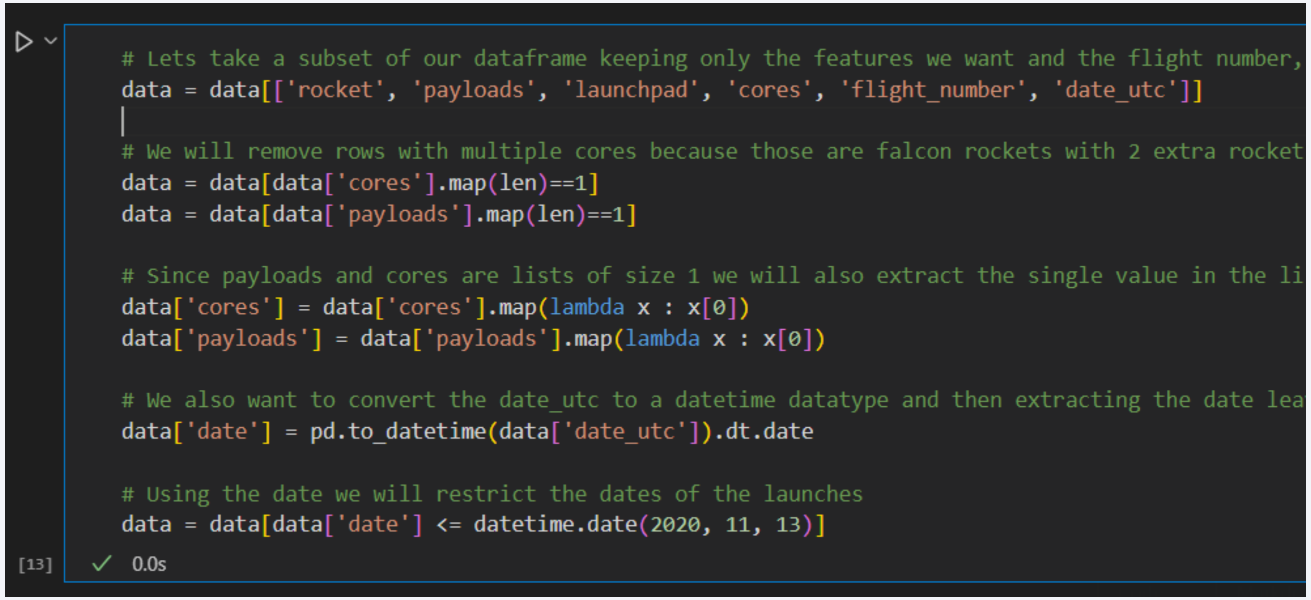
```
response = requests.get(spacex_url)
```

Use json_normalize method to convert json result to data frame



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

Performed data cleaning and filling the missing value

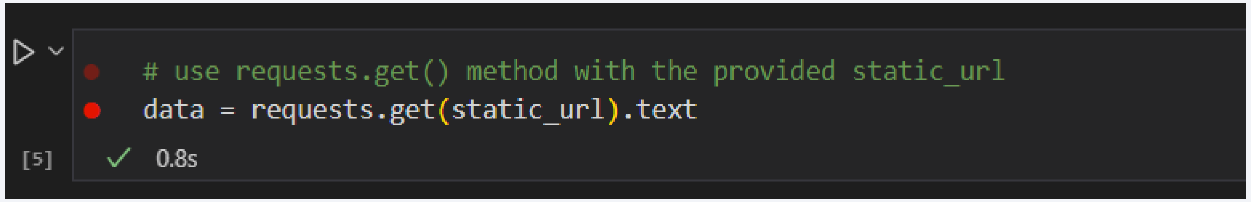


```
# Lets take a subset of our dataframe keeping only the features we want and the flight number,
data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc']]

# We will remove rows with multiple cores because those are falcon rockets with 2 extra rocket
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 li
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 lea
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)]
```

https://github.com/Mahmoudalmohammad/Data-Science-Capstone-SpaceX/blob/main/Data_Collection.ipynb

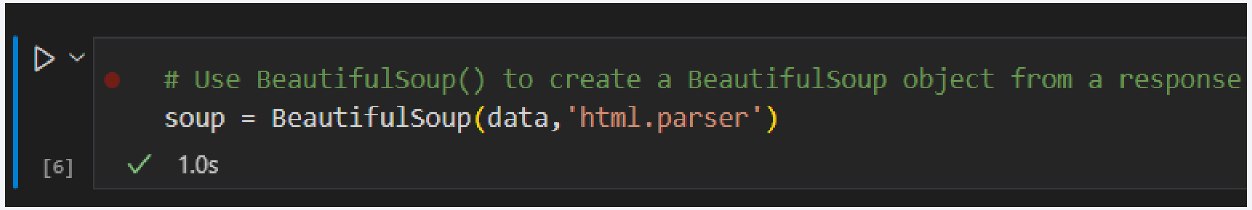
Data Collection - Scraping

Request the Falcon9 Launch Wiki page from url



```
# use requests.get() method with the provided static_url
data = requests.get(static_url).text
```

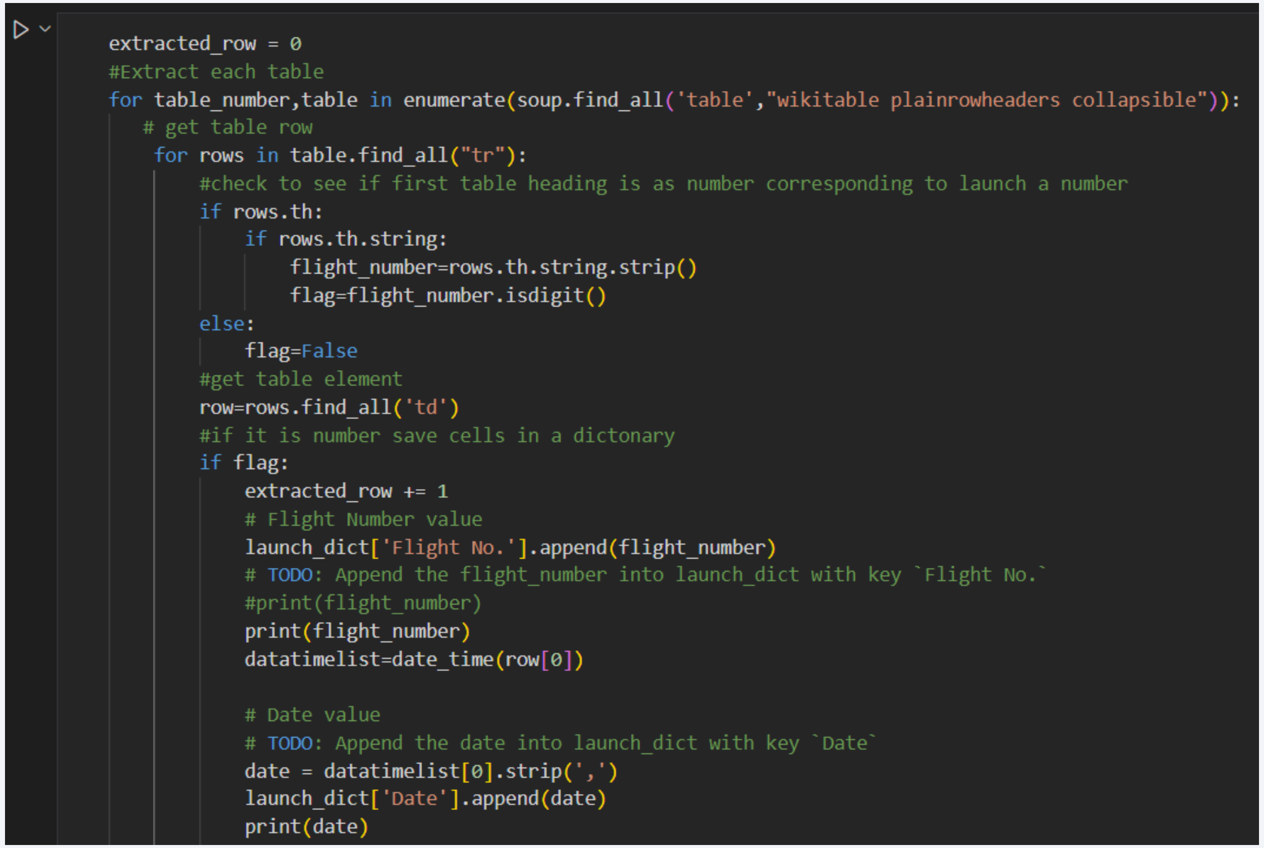
Create a BeautifulSoup from the HTML response



```
# Use BeautifulSoup() to create a BeautifulSoup object from a response
soup = BeautifulSoup(data,'html.parser')
```

Data Collection - Scraping

Extract all column/variable names from the HTML header



```python
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowheaders collapsible")):
    # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number corresponding to launch a number
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
        else:
            flag=False
        #get table element
        row=rows.find_all('td')
        #if it is number save cells in a dictonary
        if flag:
            extracted_row += 1
            # Flight Number value
            launch_dict['Flight No.'].append(flight_number)
            # TODO: Append the flight_number into launch_dict with key `Flight No.`
            #print(flight_number)
            print(flight_number)
            datatimelist=date_time(row[0])

            # Date value
            # TODO: Append the date into launch_dict with key `Date`
            date = datatimelist[0].strip(',')
            launch_dict['Date'].append(date)
            print(date)
```

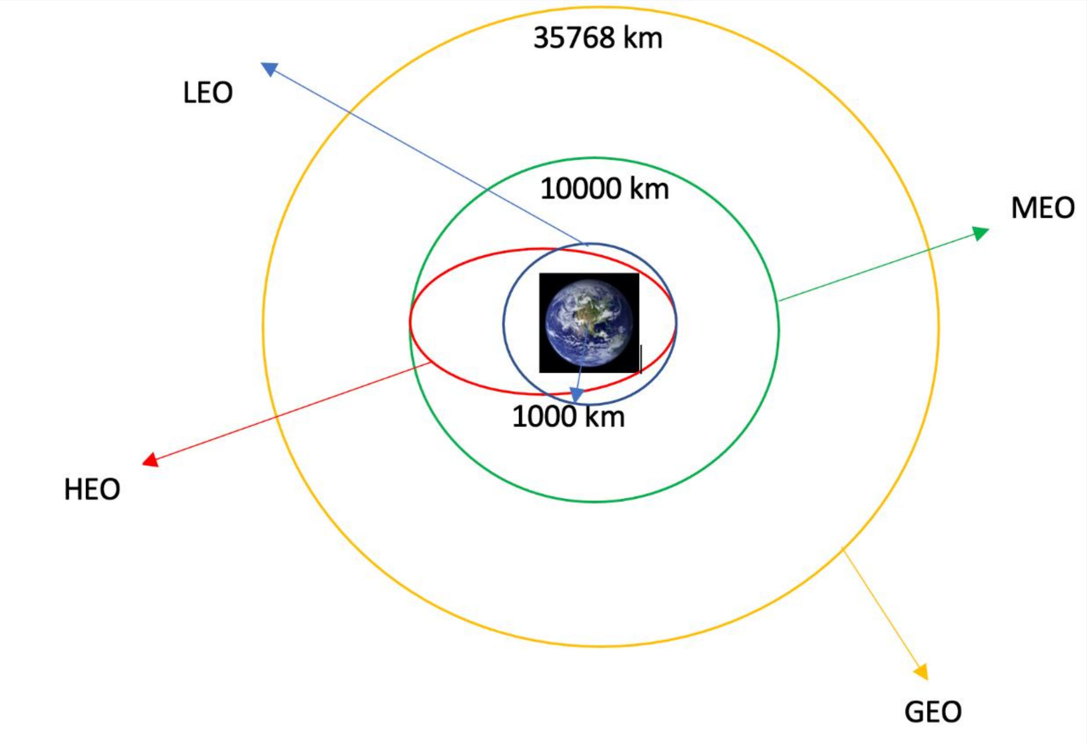
https://github.com/Mahmoudalmohammad/Data-Science-Capstone-SpaceX/blob/main/Data_Collection_with_Web_Scraping.ipynb

Data Wrangling

Data wrangling involves the cleaning and consolidation of disorderly and intricate datasets to facilitate easy access and exploratory data analysis (EDA).

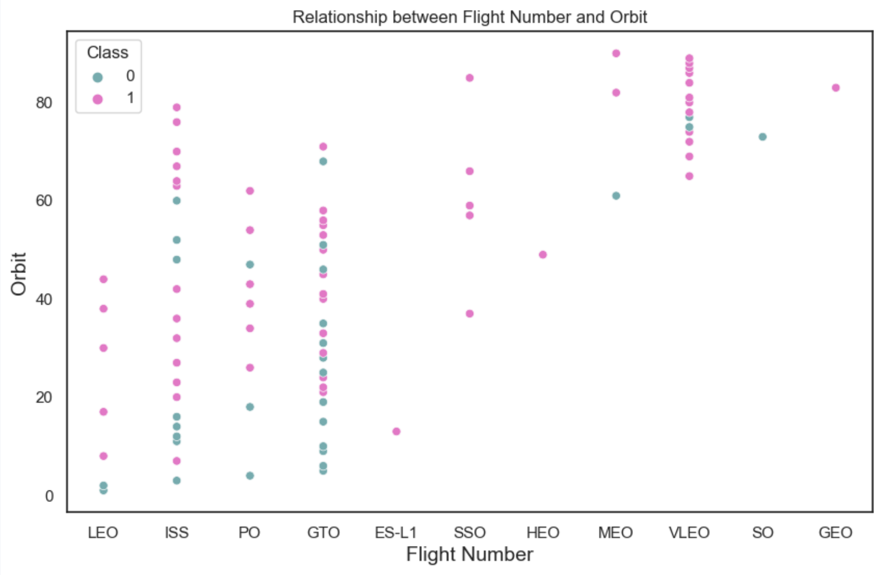
Initially, we will compute the count of launches per site, followed by determining the quantity and frequency of mission outcomes for each orbit type.

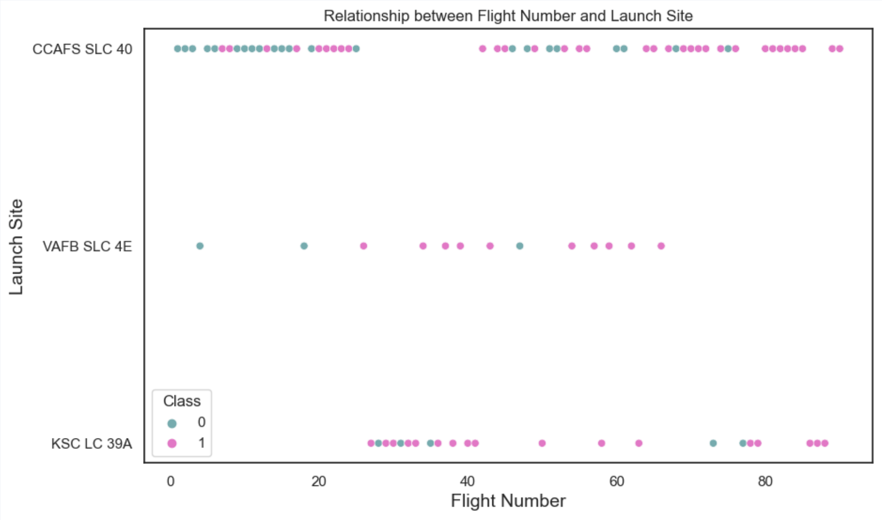
Next, we generate a landing outcome label based on the outcome column, facilitating subsequent analysis, visualization, and machine learning processes. Finally, we export the outcome to a CSV file for further use.



https://github.com/Mahmoudalmohammad/Data-Science-Capstone-SpaceX/blob/main/Data_Wrangling.ipynb

EDA with Data Visualization





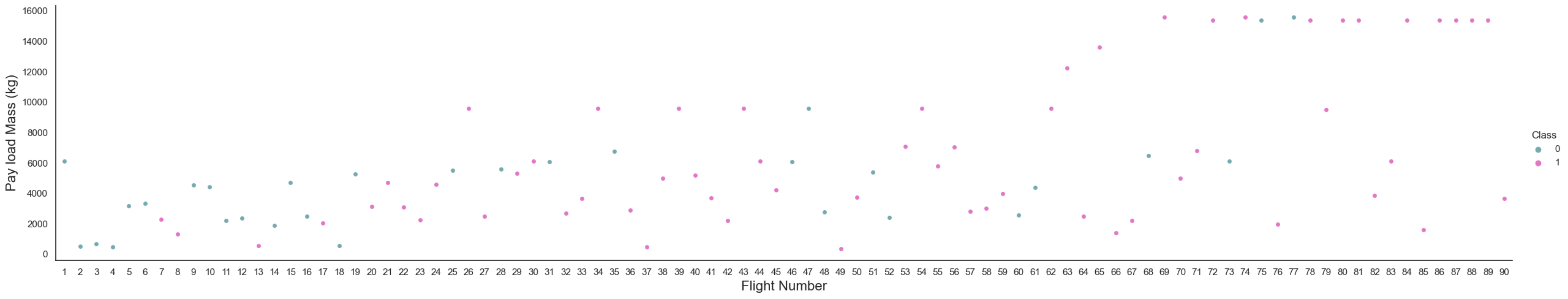
We first started by using scatter graph to find the relationship between the attributes such as between:

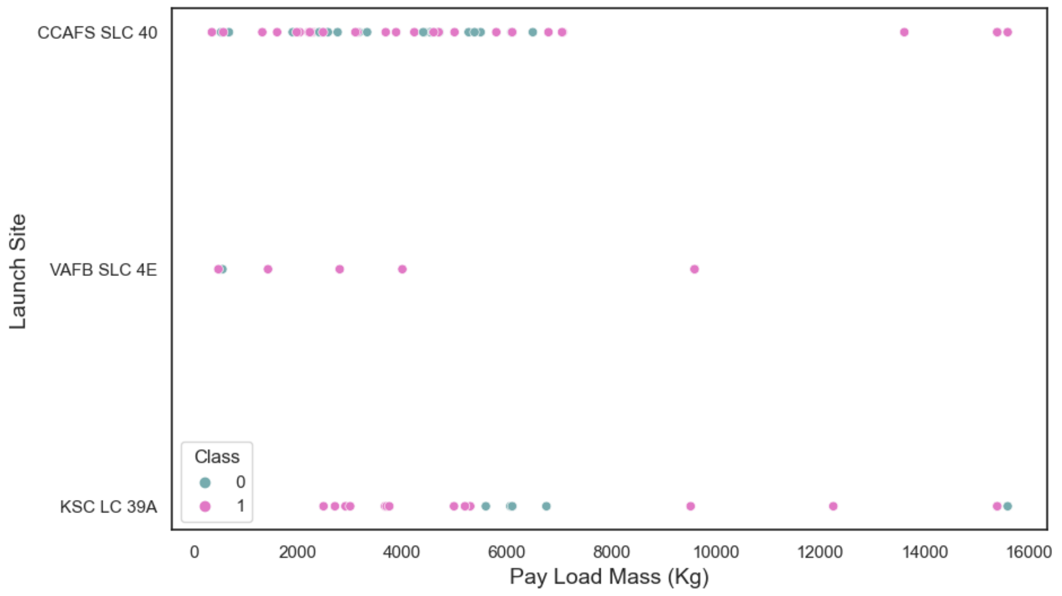
- Payload and Flight Number.
- Flight Number and Launch Site.
- Payload and Launch Site.
- Flight Number and Orbit Type.
- Payload and Orbit Type.

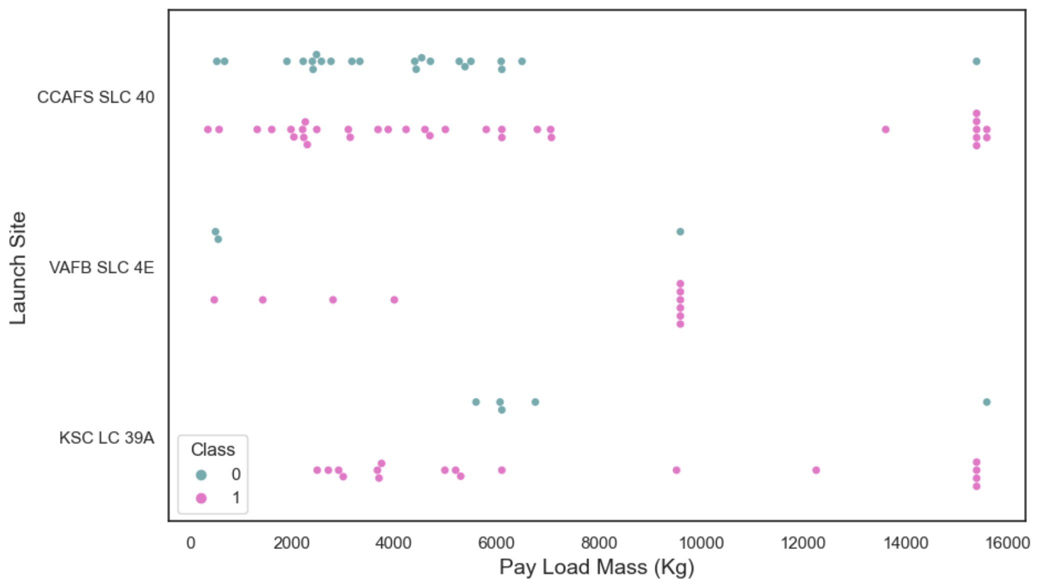
Scatter plots show dependency of attributes on each other. Once a pattern is determined from the graphs. It’s very easy to see which factors affecting the most to the success of the landing outcomes.

https://github.com/Mahmoudalmohammad/Data-Science-Capstone-SpaceX/blob/main/Exploratory_Data_Analysis_with_Visualisation.ipynb

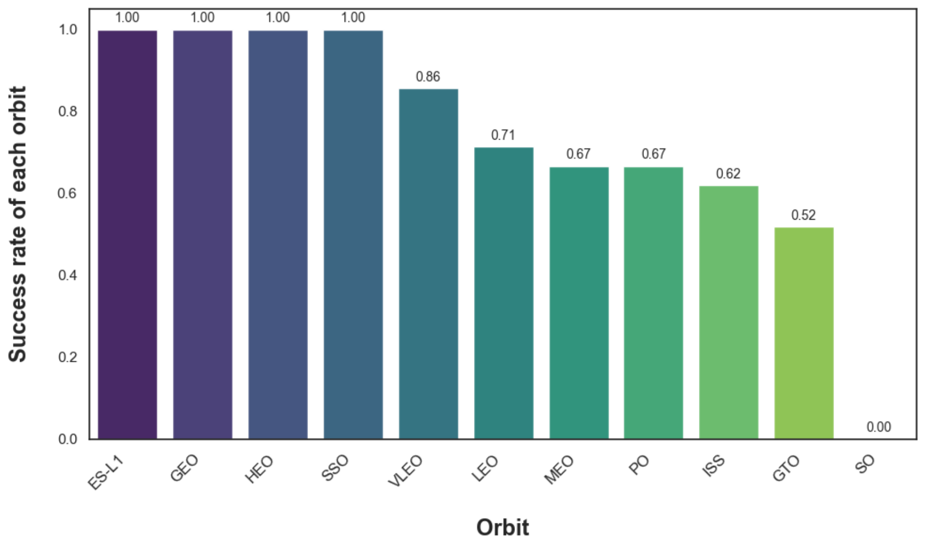
EDA with Data Visualization

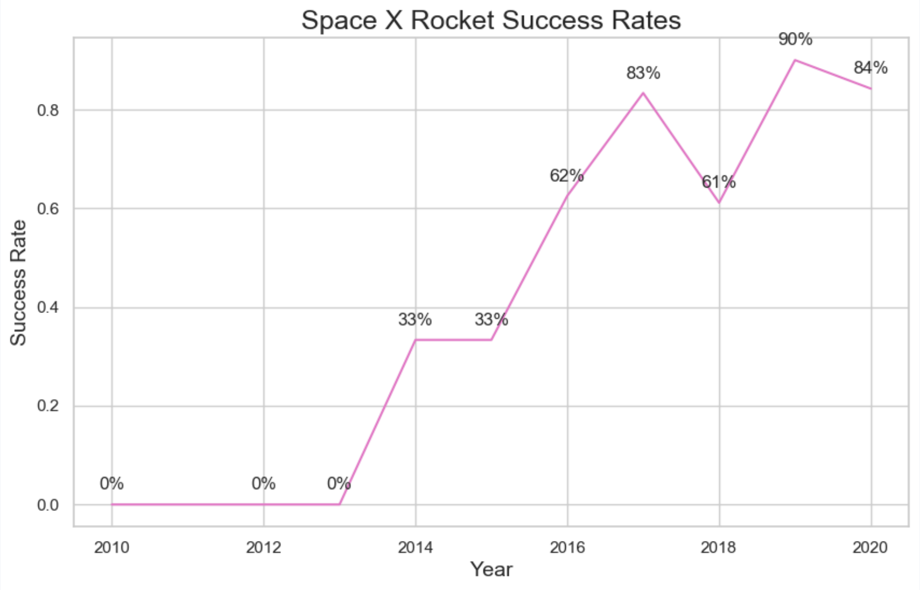






EDA with Data Visualization





After initially examining relationships with scatter plots, we employ additional visualization techniques such as bar graphs and line plots for deeper analysis.

Bar graphs provide a straightforward method to interpret attribute relationships, helping us identify which orbits hold the highest probability of success. Following this, we utilize line graphs to depict trends or patterns of attributes over time, particularly focusing on the annual trends in launch success. Subsequently, we leverage Feature Engineering to enhance future success prediction modules by creating dummy variables from categorical columns.

https://github.com/Mahmoudalmohammad/Data-Science-Capstone-SpaceX/blob/main/Exploratory_Data_Analysis_with_Visualisation.ipynb

EDA with SQL

Using SQL, we had performed many queries to get better understanding of the dataset, Ex:

- Displaying the names of the launch sites.
- Displaying 5 records where launch sites begin with the string ‘CCA’.
- Displaying the total payload mass carried by booster launched by NASA (CRS).
- Displaying the average payload mass carried by booster version F9 v1.1.
- Listing the date when the first successful landing outcome in ground pad was achieved.
- Listing the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000.

EDA with SQL

Using SQL, we had performed many queries to get better understanding of the dataset, Ex:

- Listing the total number of successful and failure mission outcomes.
- Listing the names of the booster_versions which have carried the maximum payload mass.
- Listing the failed landing_outcomes in drone ship, their booster versions, and launch sites names for in year 2015.- Rank the count of landing outcomes or success between the date 2010-06-04 and 2017-03-20, in descending order

- https://github.com/Mahmoudalmohammad/Data-Science-Capstone-SpaceX/blob/main/Exploratory_Data_Analysis_with_SQL.ipynb

Build an Interactive Map with Folium

To create an interactive map visualizing launch data, we first obtained latitude and longitude coordinates for each launch site and placed a circle marker around each site, labeling them with their respective names.
Next, we categorized the launch outcomes (success or failure) into classes 0 and 1, representing them with Red and Green markers on the map, utilizing MarkerCluster() for clustering.

Subsequently, we employed Haversine's formula to compute the distances between the launch sites and various landmarks, aiming to address inquiries such as:

•Proximity of launch sites to railways, highways, and coastlines.
•Distance between launch sites and nearby cities.

https://github.com/Mahmoudalmohammad/Data-Science-Capstone-SpaceX/blob/main/Interactive_Visual_Analytics_with_Folium.ipynb

Build a Dashboard with Plotly Dash

We developed an interactive dashboard using Plotly Dash, enabling users to manipulate the data according to their preferences. We depicted pie charts illustrating the total launches per specific sites, and subsequently generated scatter plots illustrating the relationship between Outcome and Payload Mass (Kg) across different booster versions.

https://github.com/Mahmoudalmohammad/Data-Science-Capstone-SpaceX/blob/main/SpaceX_Dash.py

Predictive Analysis (Classification)

Building the Model

- Load the dataset into NumPy and Pandas
- Transform the data and then split into training and test datasets
- Decide which type of ML to use
- set the parameters and algorithms to GridSearchCV and fit it to dataset.

Evaluating the Model

- Check the accuracy for each model
- Get tuned hyperparameters for each type of algorithms.
- plot the confusion matrix.

Improving the Model

- Use Feature Engineering and Algorithm Tuning

Find the Best Model

- The model with the best accuracy score will be the best performing model.

https://github.com/Mahmoudalmohammad/Data-Science-Capstone-SpaceX/blob/main/SpaceX_Dash.py

Results

The results will be categorized to 3 main results which is:

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

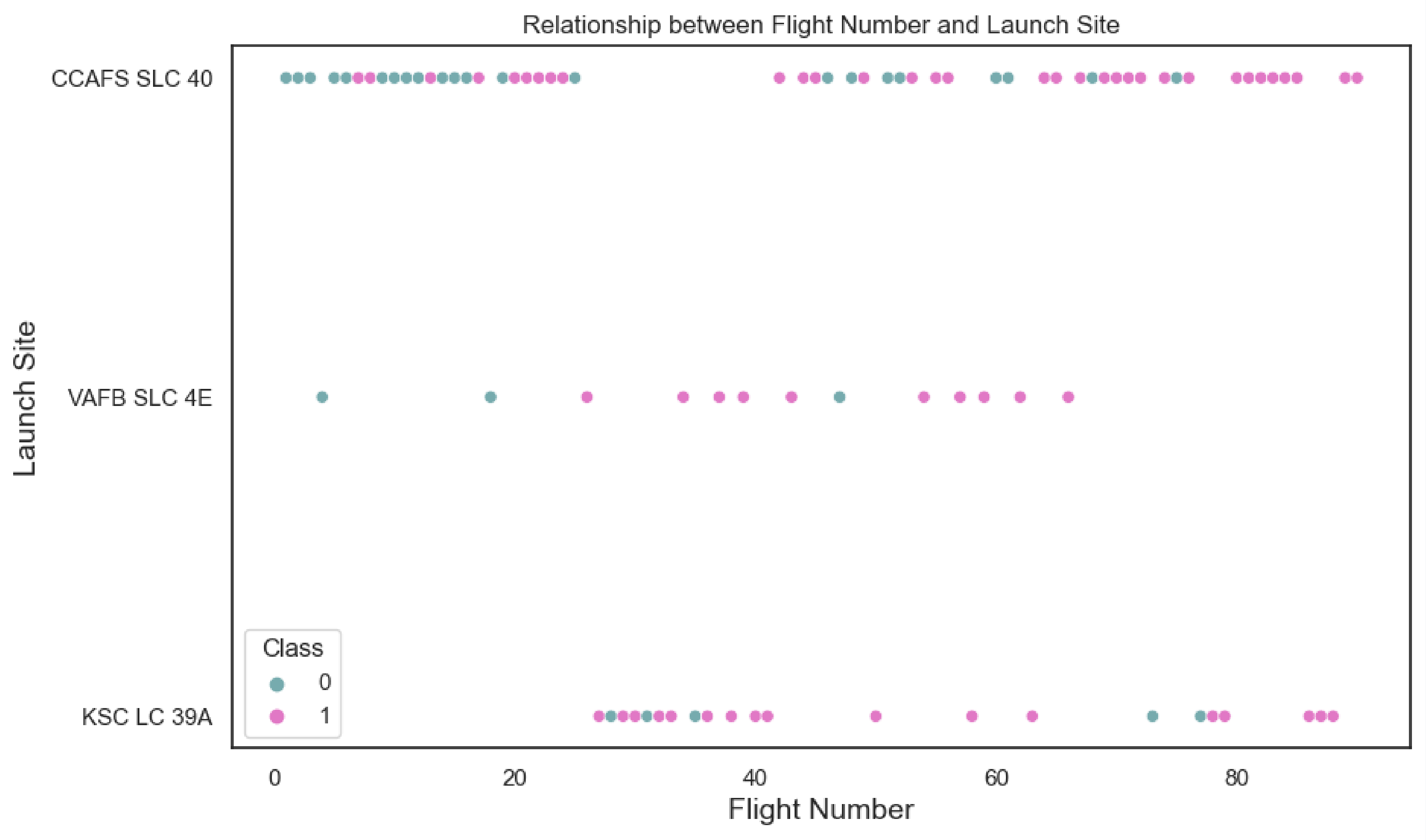
Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

This scatter plot demonstrates that as the number of flights from the launch site increases, the success rate also tends to rise accordingly.

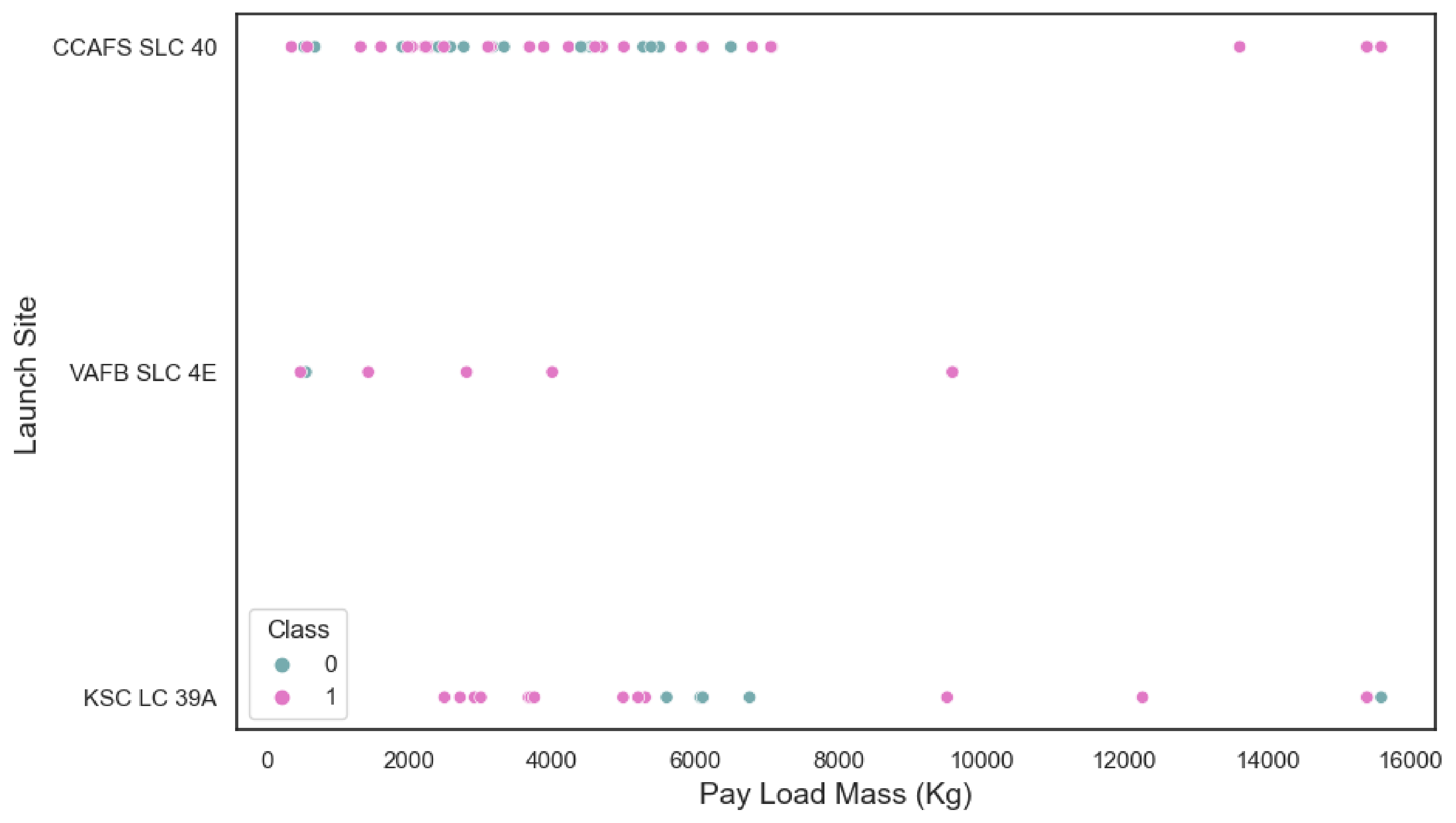
Nevertheless, the pattern exhibited at site CCAFS SLC40 is the least pronounced.



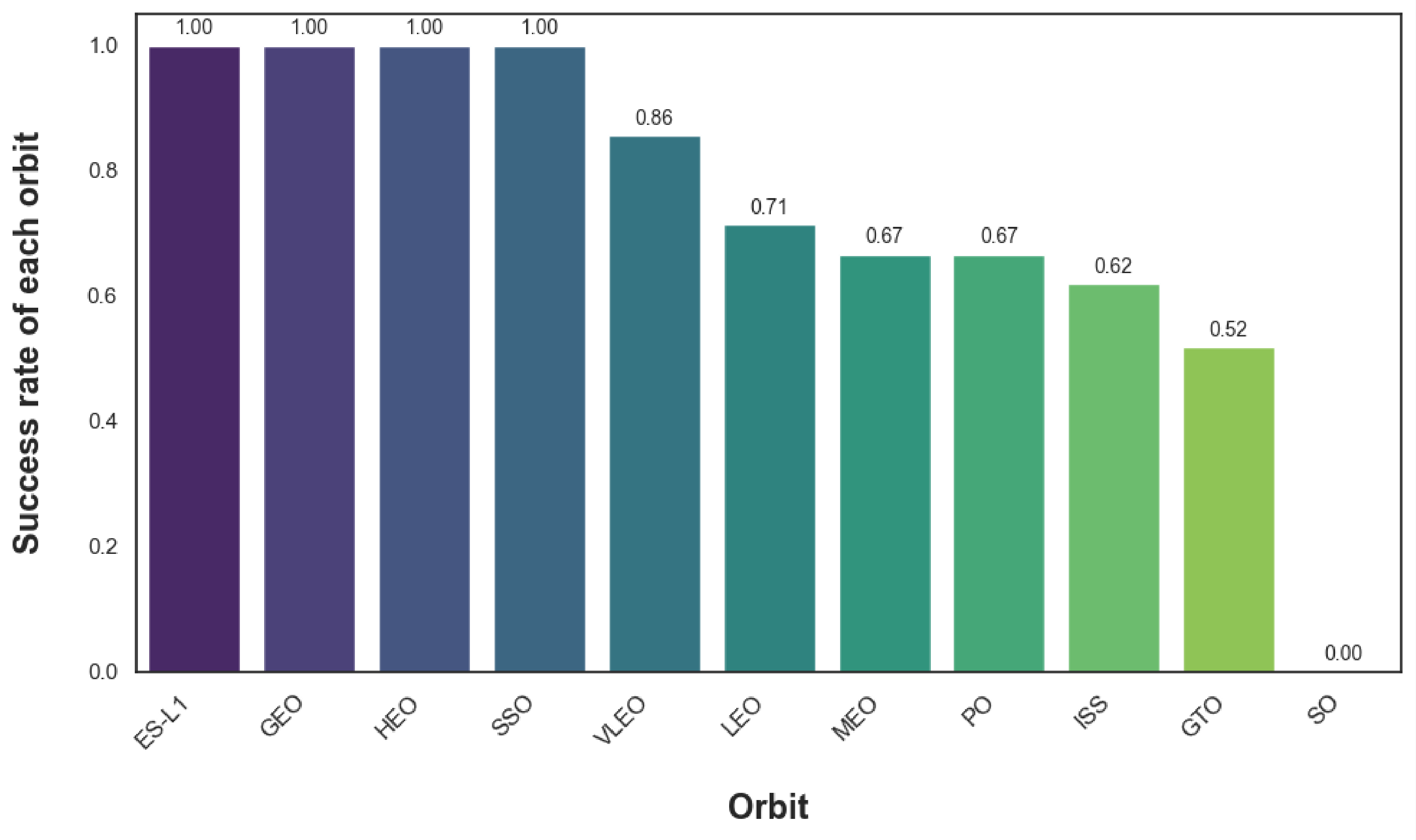
Payload vs. Launch Site

The scatter plot illustrates that when the payload mass exceeds 7000kg, there is a substantial increase in the probability of success.

However, no discernible pattern indicates that the launch site is dependent on the payload mass for the success rate.



Success Rate vs. Orbit Type

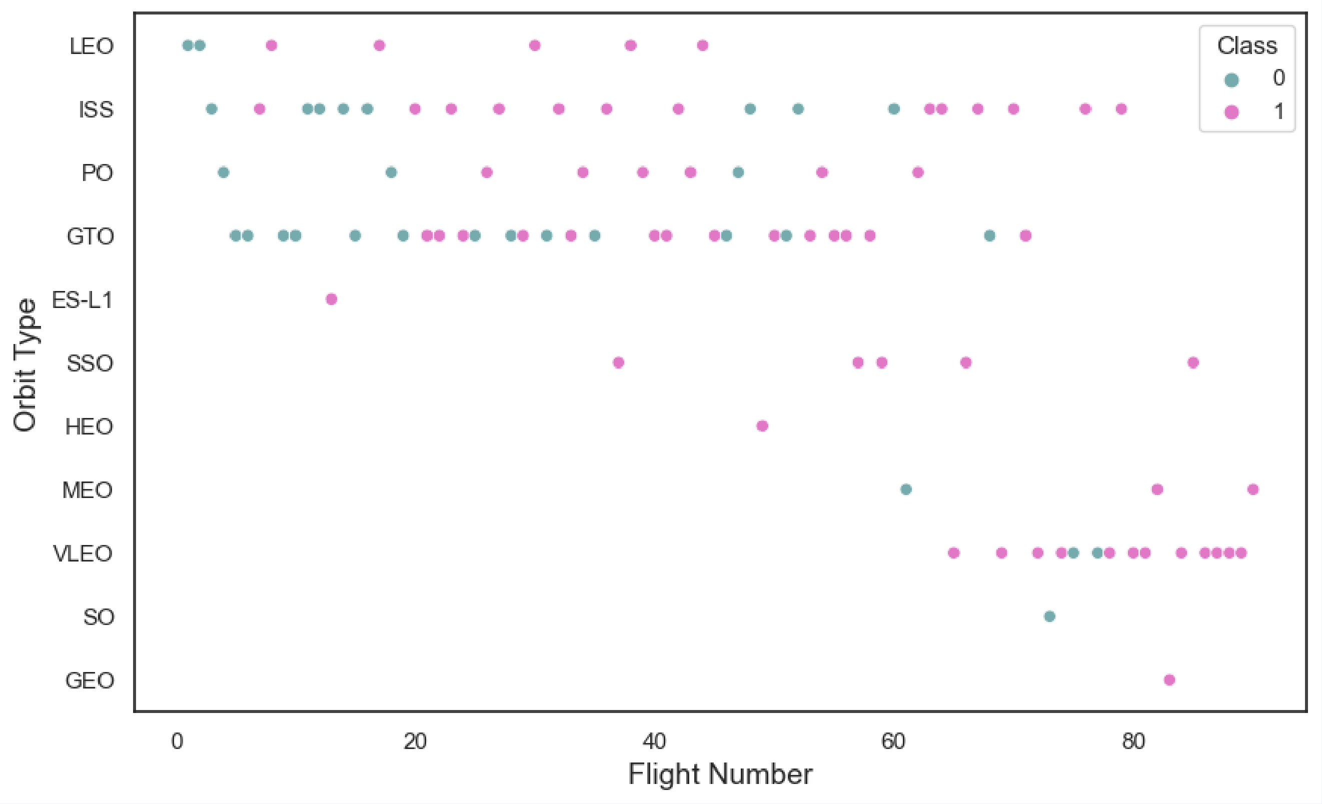


The illustration suggests that the various orbits can significantly impact landing outcomes, with certain orbits like SSO, HEO, GEO, and ES-L1 boasting a 100% success rate, while the SO orbit shows a 0% success rate.

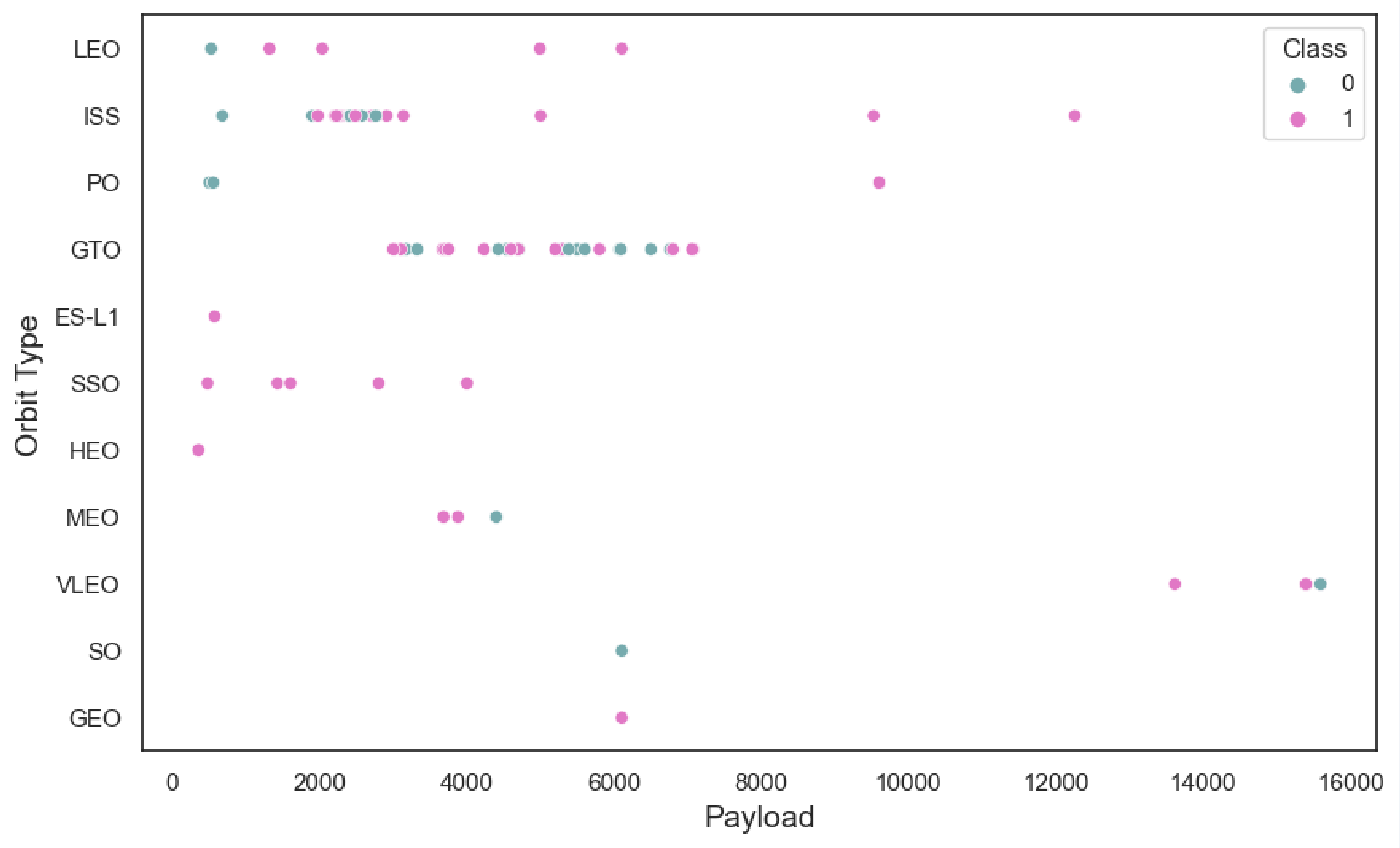
Upon closer examination, it becomes evident that some orbits, including GEO, SO, HEO, and ES-L1, only have one occurrence each. Consequently, this dataset lacks sufficient data points to discern patterns or trends, necessitating further data collection before drawing any conclusions.

Flight Number vs. Orbit Type

The scatter plot indicates a trend where higher flight numbers within each orbit correlate with higher success rates, particularly noticeable in LEO orbits. However, the relationship between flight number and success rate is inconclusive for GTO orbits. It's important to exclude orbits with only one occurrence from the analysis due to insufficient data.



Payload vs. Orbit Type

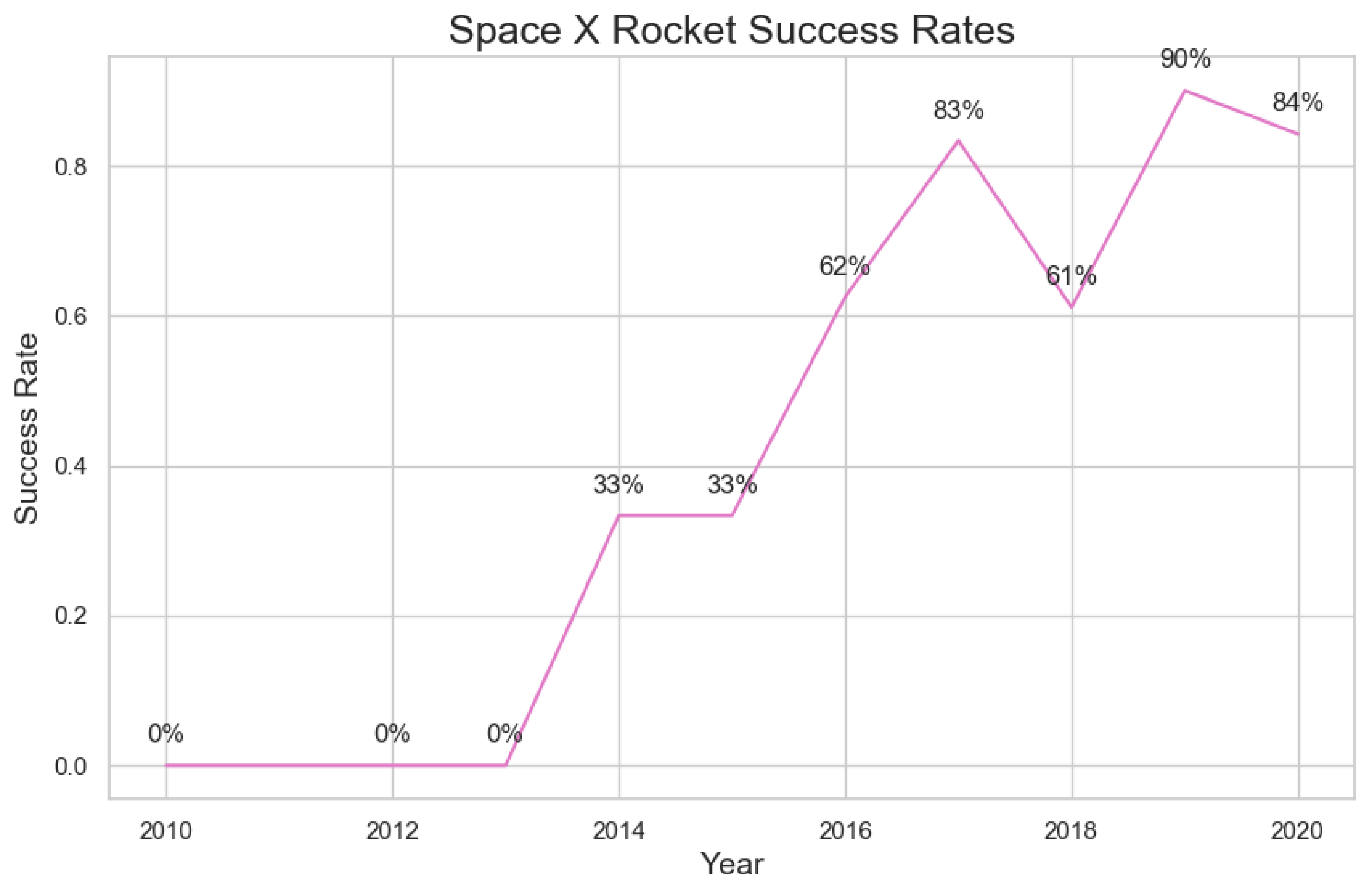


A heavier payload benefits Low Earth Orbit (LEO), the International Space Station (ISS), and Polar Orbit (P0), but it adversely affects Medium Earth Orbit (MEO) and Very Low Earth Orbit (VLEO).

The Geostationary Transfer Orbit (GTO) shows no discernible relationship between the attributes. However, for Sun-Synchronous Orbit (SO), Geostationary Orbit (GEO), and Highly Elliptical Orbit (HEO), further data is required to identify any patterns or trends.

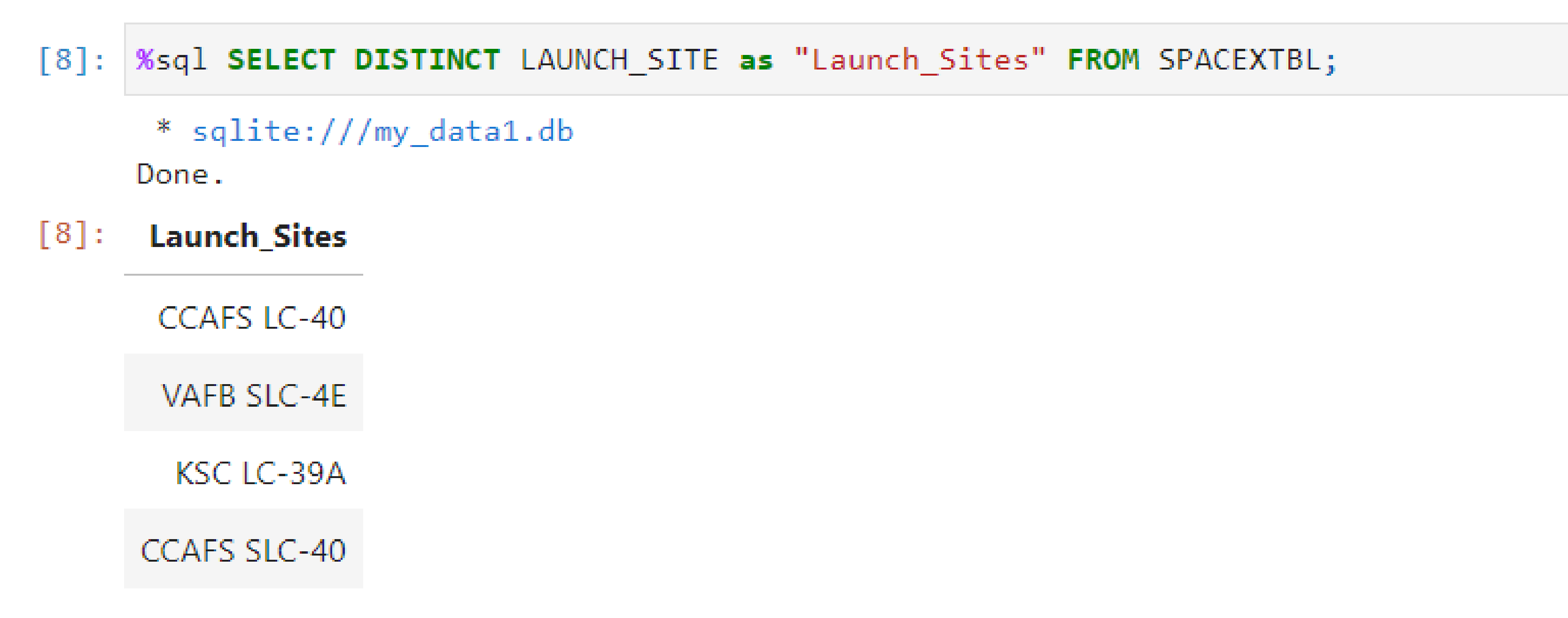
Launch Success Yearly Trend

These figures unmistakably illustrate a rising trend from 2013 to 2020. If this trend persists beyond the next year, the success rate will steadily climb until it reaches 1/100%.



All Launch Site Names

We used the key word **DISTINCT** to show only unique launch sites from the SpaceX data.



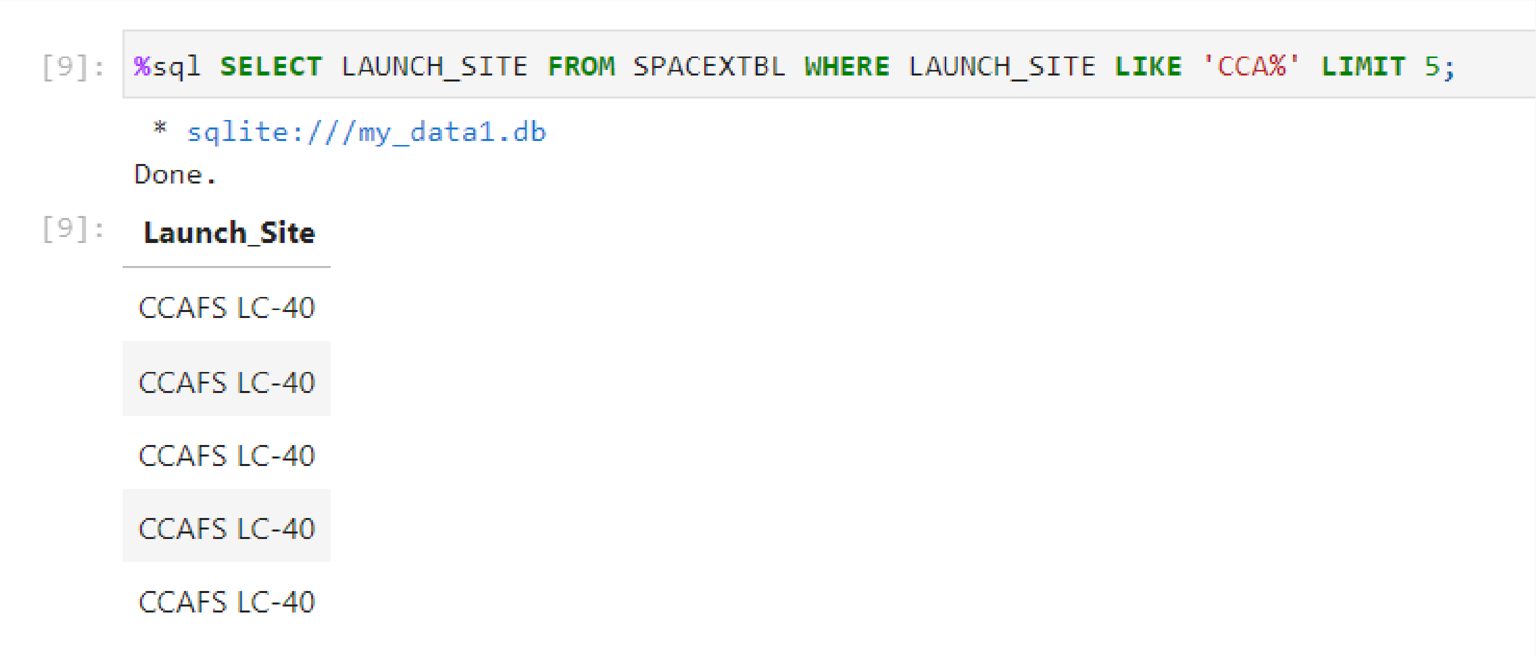
```
[8]: %sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL;

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

[8]:

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

Launch Site Names Begin with 'CCA'

We used the query above to display 5 records where launch sites begin with `CCA`



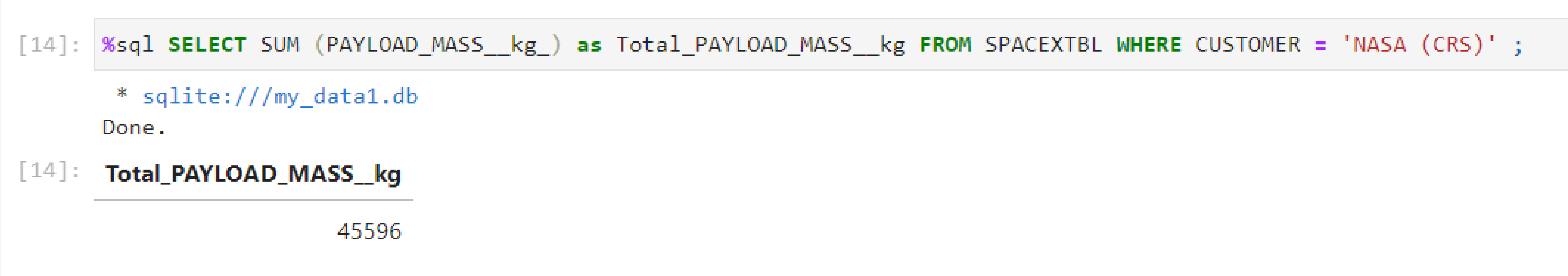
```
[9]: %sql SELECT LAUNCH_SITE FROM SPACEXTBL WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5;

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

[9]:

Launch_Site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

Total Payload Mass

We calculated the total payload carried by boosters from NASA as 45596 using the query below



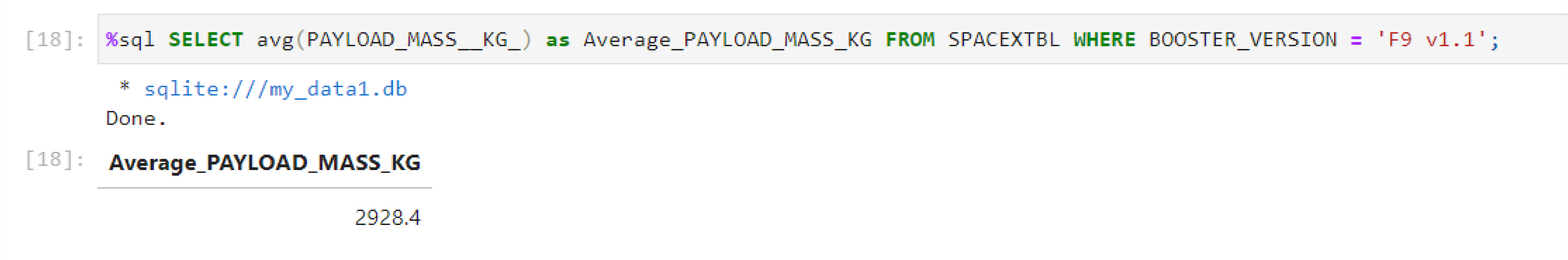
```
[14]: %sql SELECT SUM (PAYLOAD_MASS__kg_) as Total_PAYLOAD_MASS__kg FROM SPACEXTBL WHERE CUSTOMER = 'NASA (CRS)' ;

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

[14]:

Total_PAYLOAD_MASS__kg
45596

Average Payload Mass by F9 v1.1

We calculated the average payload mass carried by booster version F9 v1.1 as 2928.4

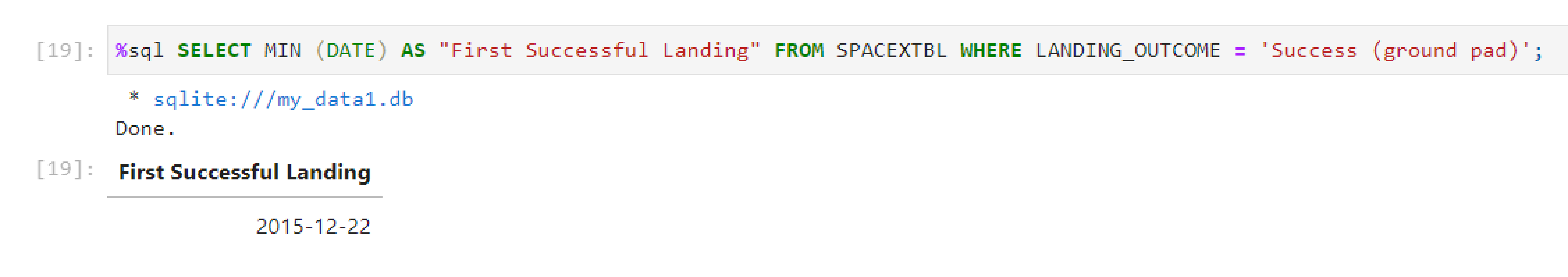


```
[18]: %sql SELECT avg(PAYLOAD_MASS__KG_) as Average_PAYLOAD_MASS_KG FROM SPACEXTBL WHERE BOOSTER_VERSION = 'F9 v1.1';

 * sqlite:///my_data1.db
Done.

[18]: Average_PAYLOAD_MASS_KG
                       2928.4
```

First Successful Ground Landing Date

We use the min() function to find the result

We observed that the dates of the first successful landing outcome on ground pad was 22nd December 2015



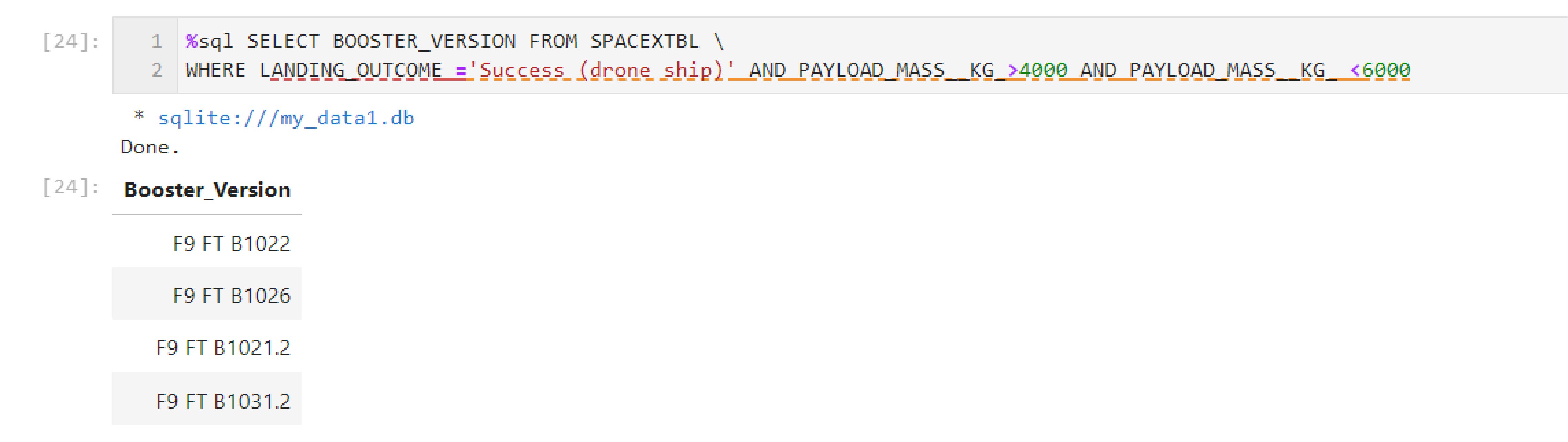
```
[19]: %sql SELECT MIN (DATE) AS "First Successful Landing" FROM SPACEXTBL WHERE LANDING_OUTCOME = 'Success (ground pad)';

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

[19]:

First Successful Landing
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

We used the **WHERE** clause to filter for boosters which have successfully landed on drone ship and applied the **AND** condition to determine successful landing with payload mass greater than 4000 but less than 6000



```
[24]: %sql SELECT BOOSTER_VERSION FROM SPACEXTBL \
      WHERE LANDING_OUTCOME ='Success (drone ship)' AND PAYLOAD_MASS__KG_ >4000 AND PAYLOAD_MASS__KG_ <6000
```

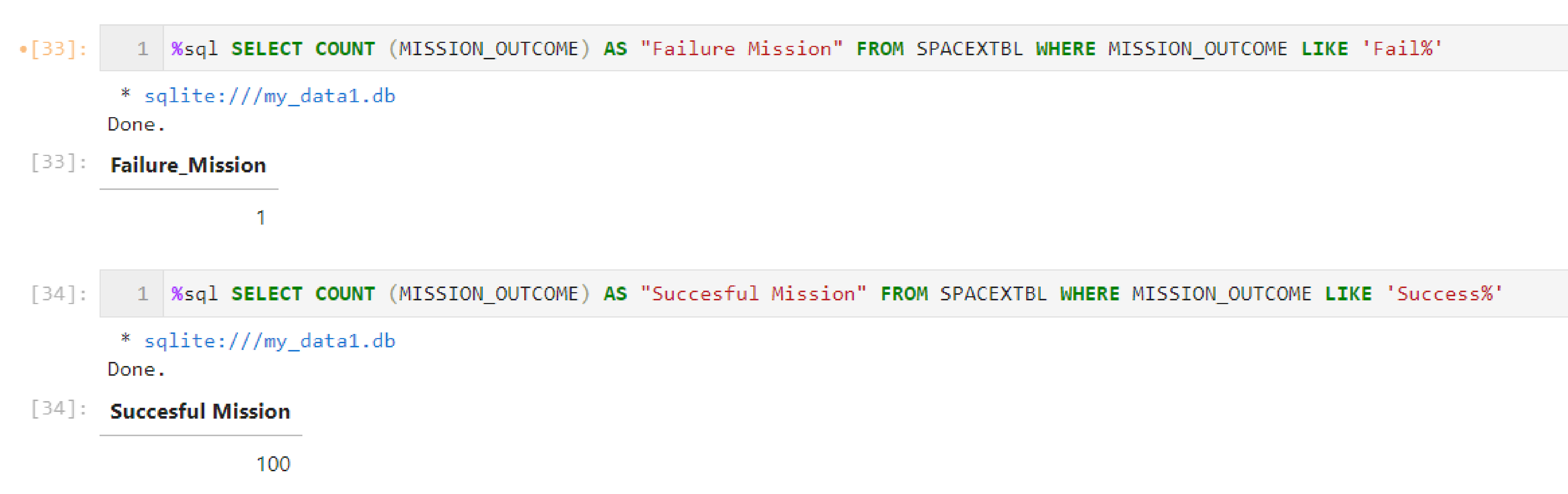
```
 * sqlite:///my_data1.db
Done.
```

[24]:

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

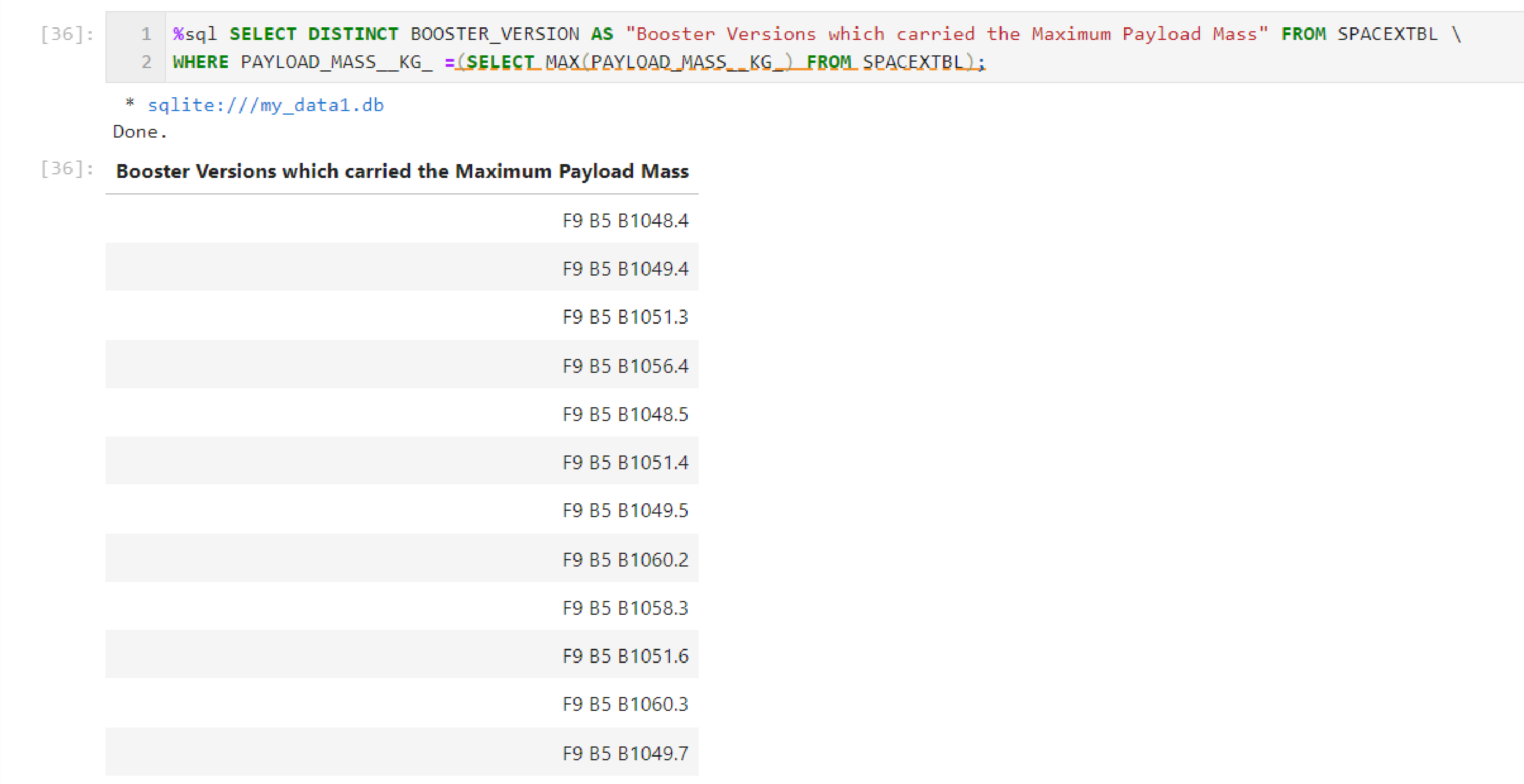
Total Number of Successful and Failure Mission Outcomes

We used wildcard like ‘%’ to filter for **WHERE** MissionOutcome was a success or a failure.



```
•[33]:  %sql SELECT COUNT (MISSION_OUTCOME) AS "Failure Mission" FROM SPACEXTBL WHERE MISSION_OUTCOME LIKE 'Fail%'

         * sqlite:///my_data1.db
        Done.

 [33]:  Failure_Mission
                      1

 [34]:  %sql SELECT COUNT (MISSION_OUTCOME) AS "Succesful Mission" FROM SPACEXTBL WHERE MISSION_OUTCOME LIKE 'Success%'

         * sqlite:///my_data1.db
        Done.

 [34]:  Succesful Mission
                      100
```

Boosters Carried Maximum Payload



```
[36]:  %sql SELECT DISTINCT BOOSTER_VERSION AS "Booster Versions which carried the Maximum Payload Mass" FROM SPACEXTBL \
       WHERE PAYLOAD_MASS__KG_ =(SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
```

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

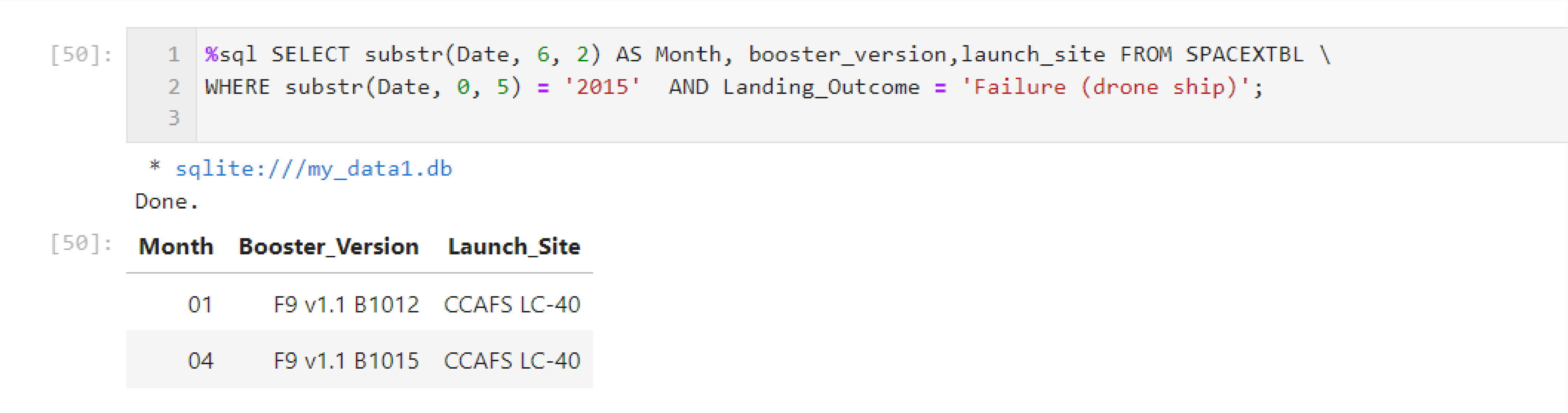
[36]:

Booster Versions which carried the Maximum Payload Mass
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

We determined the booster that have carried the maximum payload using a subquery in the **WHERE** clause and the **MAX()** function.

2015 Launch Records

We used substr(Date, 6,2) as month to get the months and substr(Date,0,5)='2015' for year and select Month, booster_version,launch_site



[50]:

```
%sql SELECT substr(Date, 6, 2) AS Month, booster_version,launch_site FROM SPACEXTBL \
WHERE substr(Date, 0, 5) = '2015'  AND Landing_Outcome = 'Failure (drone ship)';
```

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

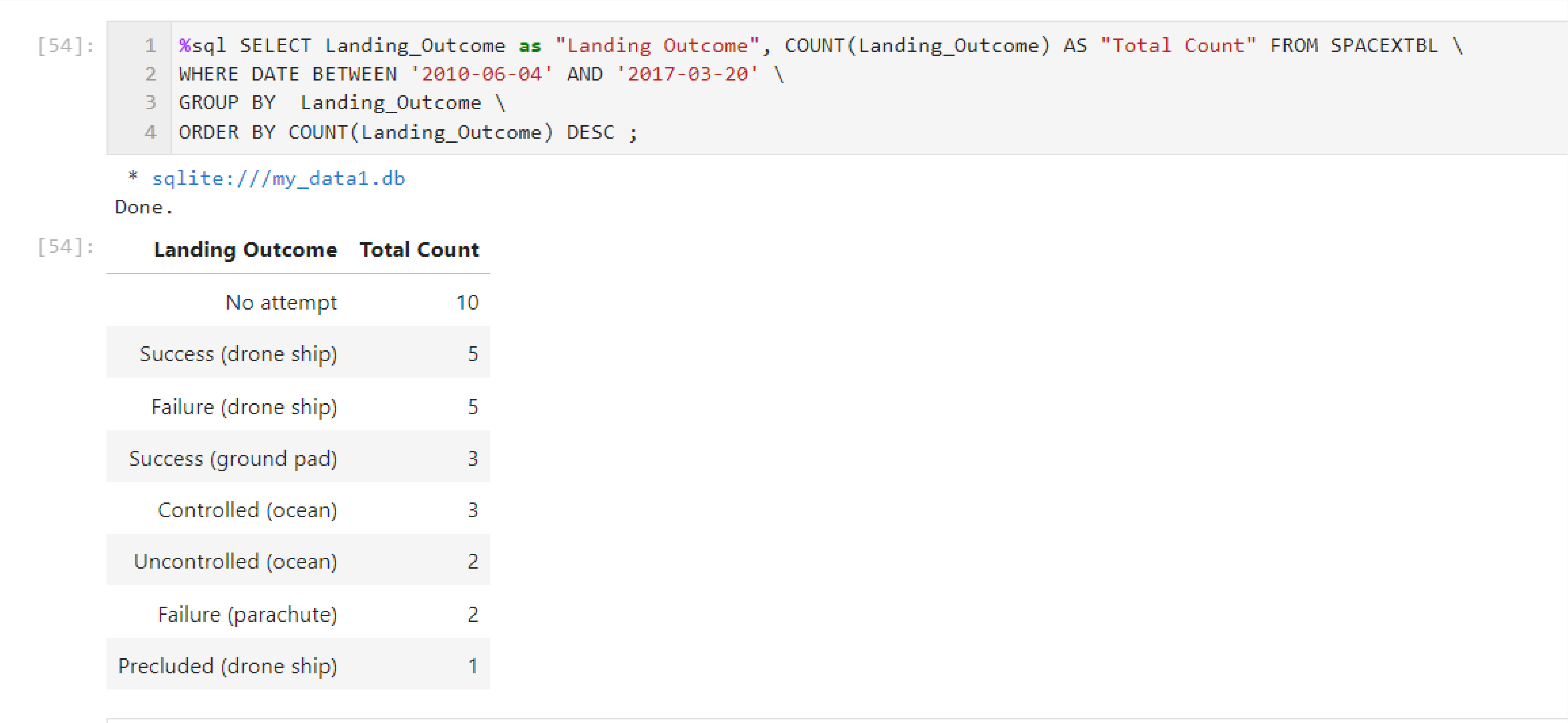
[50]:

Month	Booster_Version	Launch_Site
01	F9 v1.1 B1012	CCAFS LC-40
04	F9 v1.1 B1015	CCAFS LC-40

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

We selected Landing outcomes and the **COUNT** of landing outcomes from the data and used the **WHERE** clause to filter for landing outcomes **BETWEEN** 2010-06-04 to 2010-03-20.

We applied the **GROUP BY** clause to group the landing outcomes and the **ORDER BY** clause to order the grouped landing outcome in descending order.



[54]:

```
%sql SELECT Landing_Outcome as "Landing Outcome", COUNT(Landing_Outcome) AS "Total Count" FROM SPACEXTBL \
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' \
GROUP BY  Landing_Outcome \
ORDER BY COUNT(Landing_Outcome) DESC ;
```

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

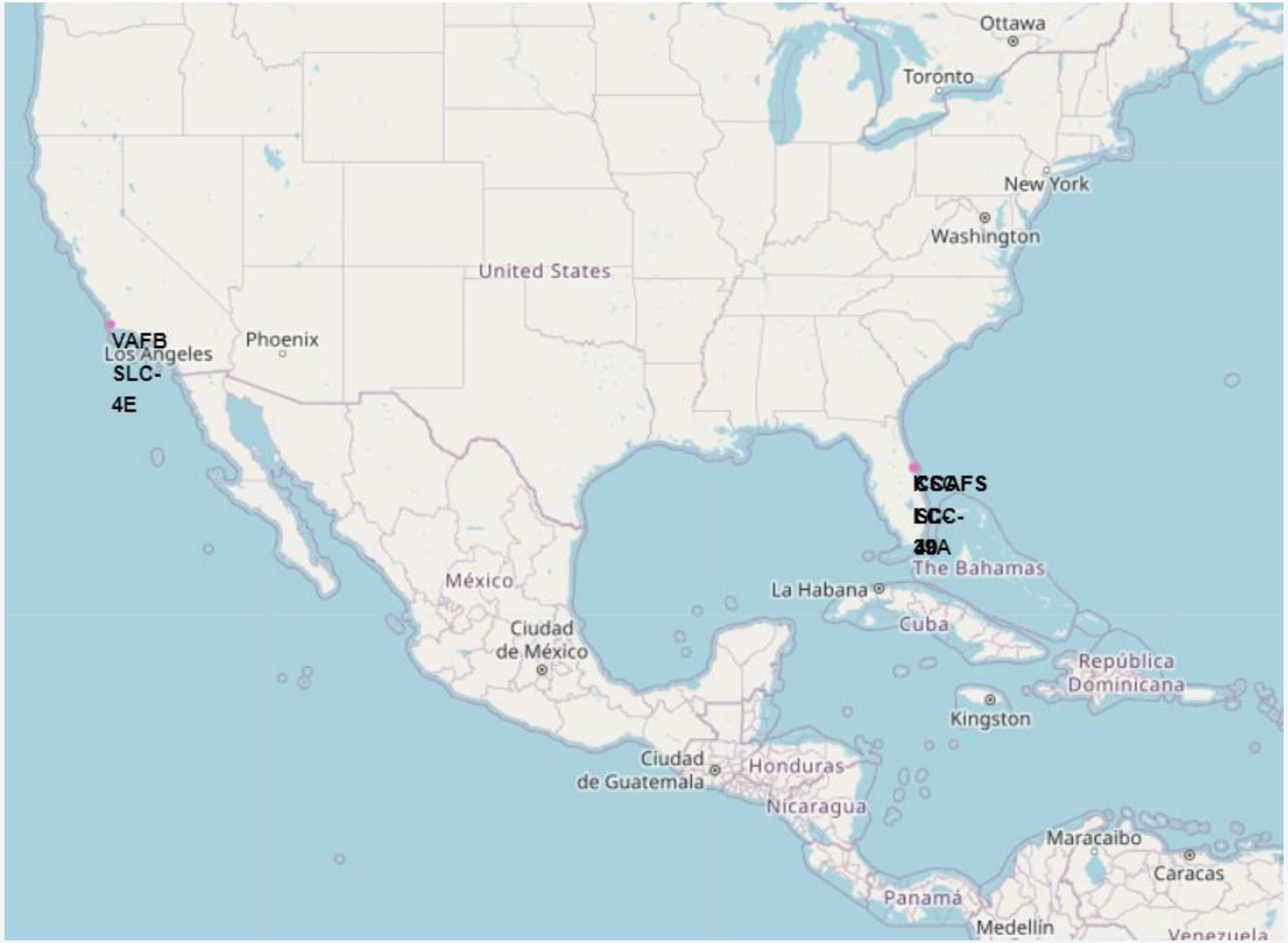
[54]:

Landing Outcome	Total Count
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Section 3

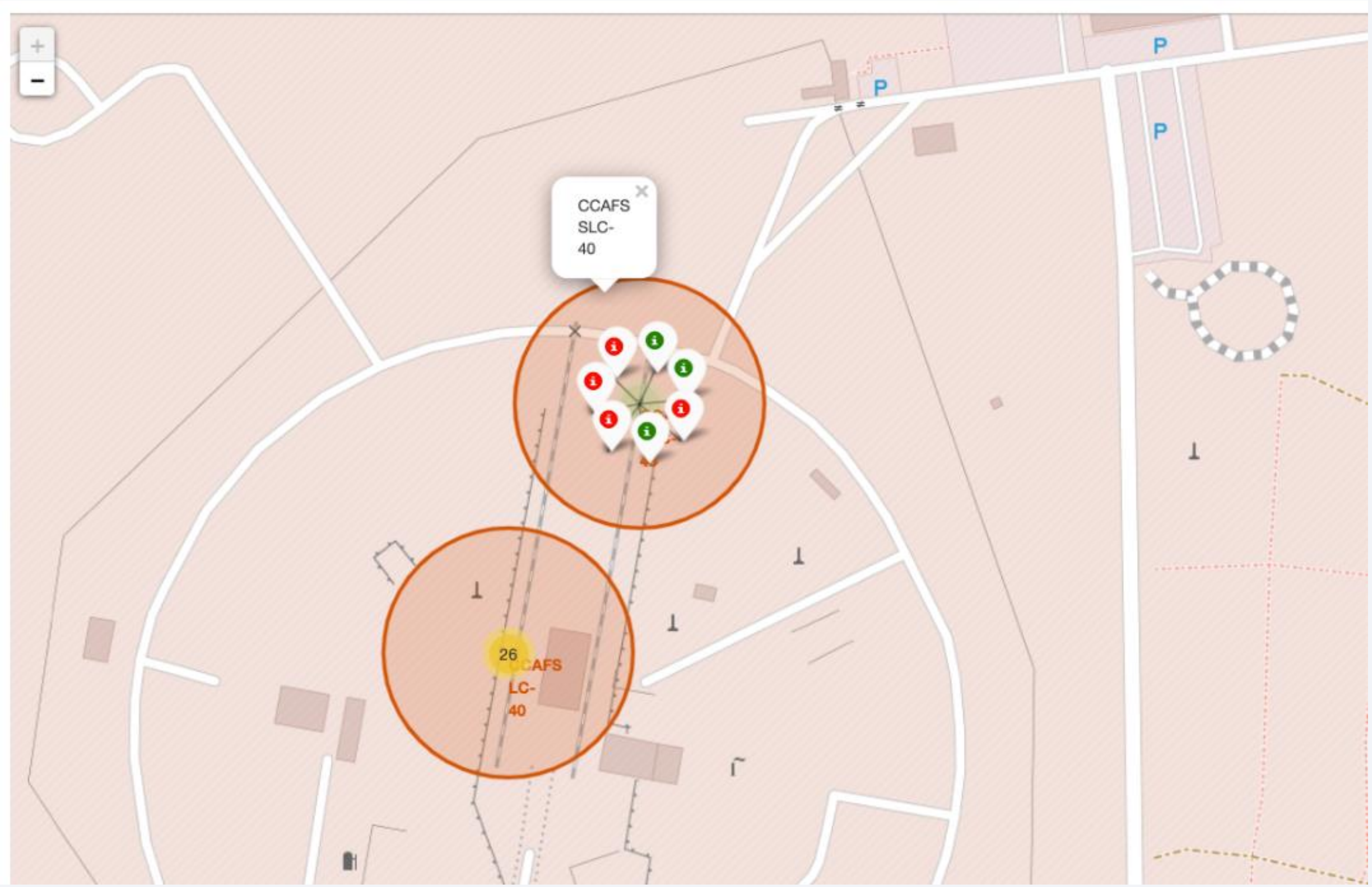
Launch Sites Proximities Analysis

Location of all the Launch Sites

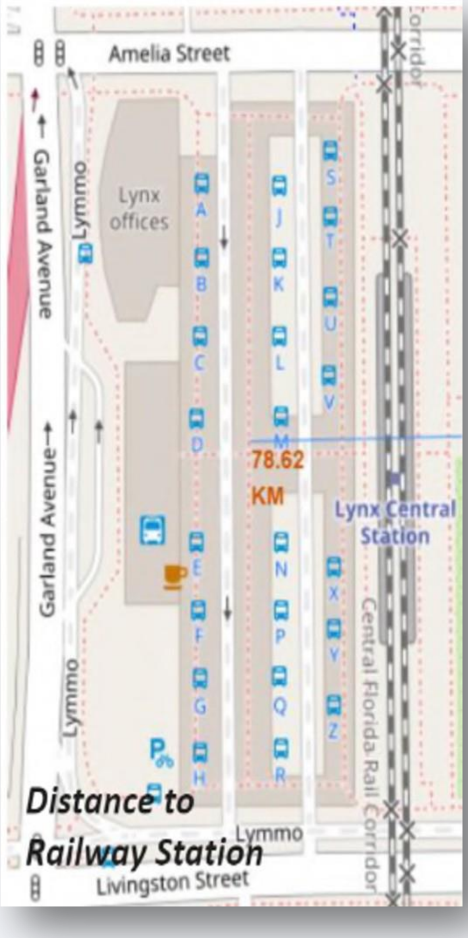


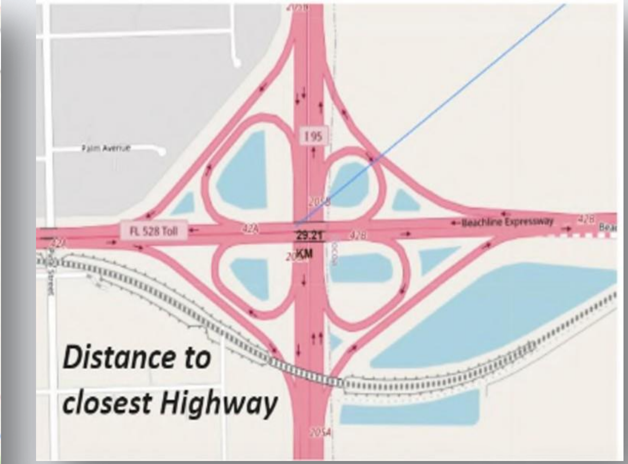
We can see that all the SpaceX launch sites are located inside the United States

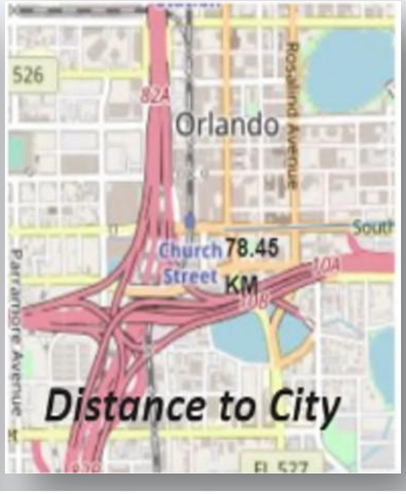
Markers showing launch sites with color labels

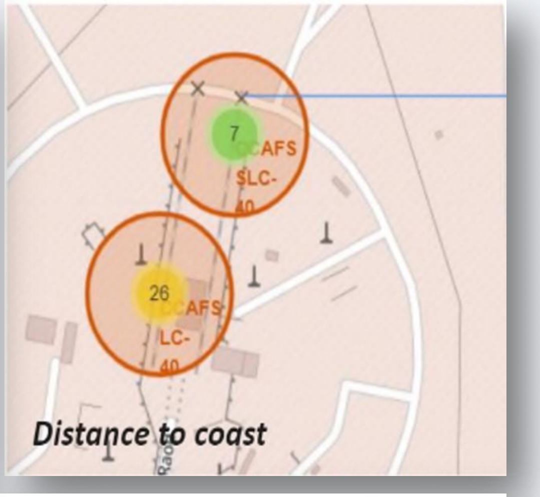


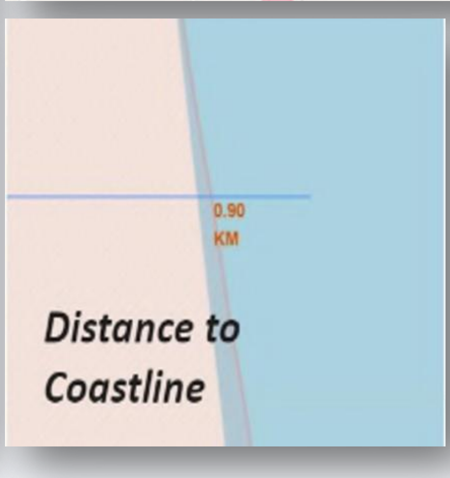
Launch Sites Distance to Landmarks

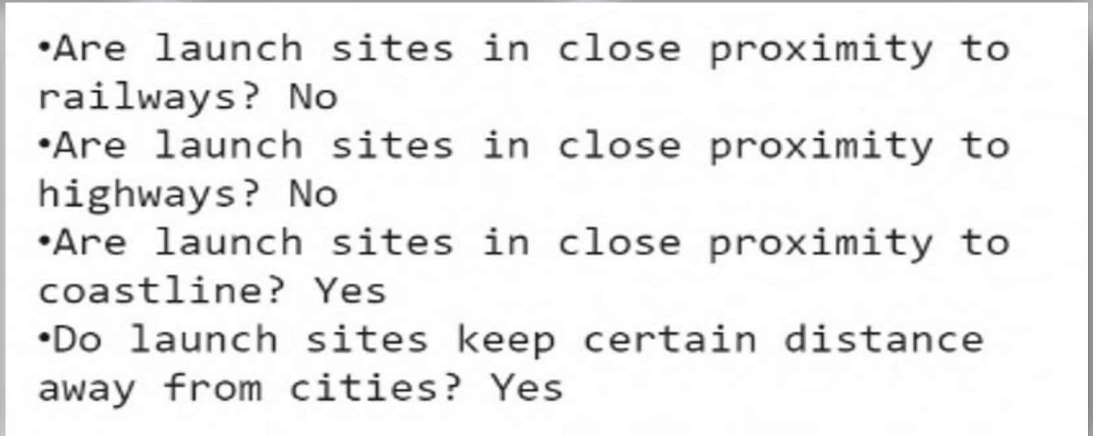




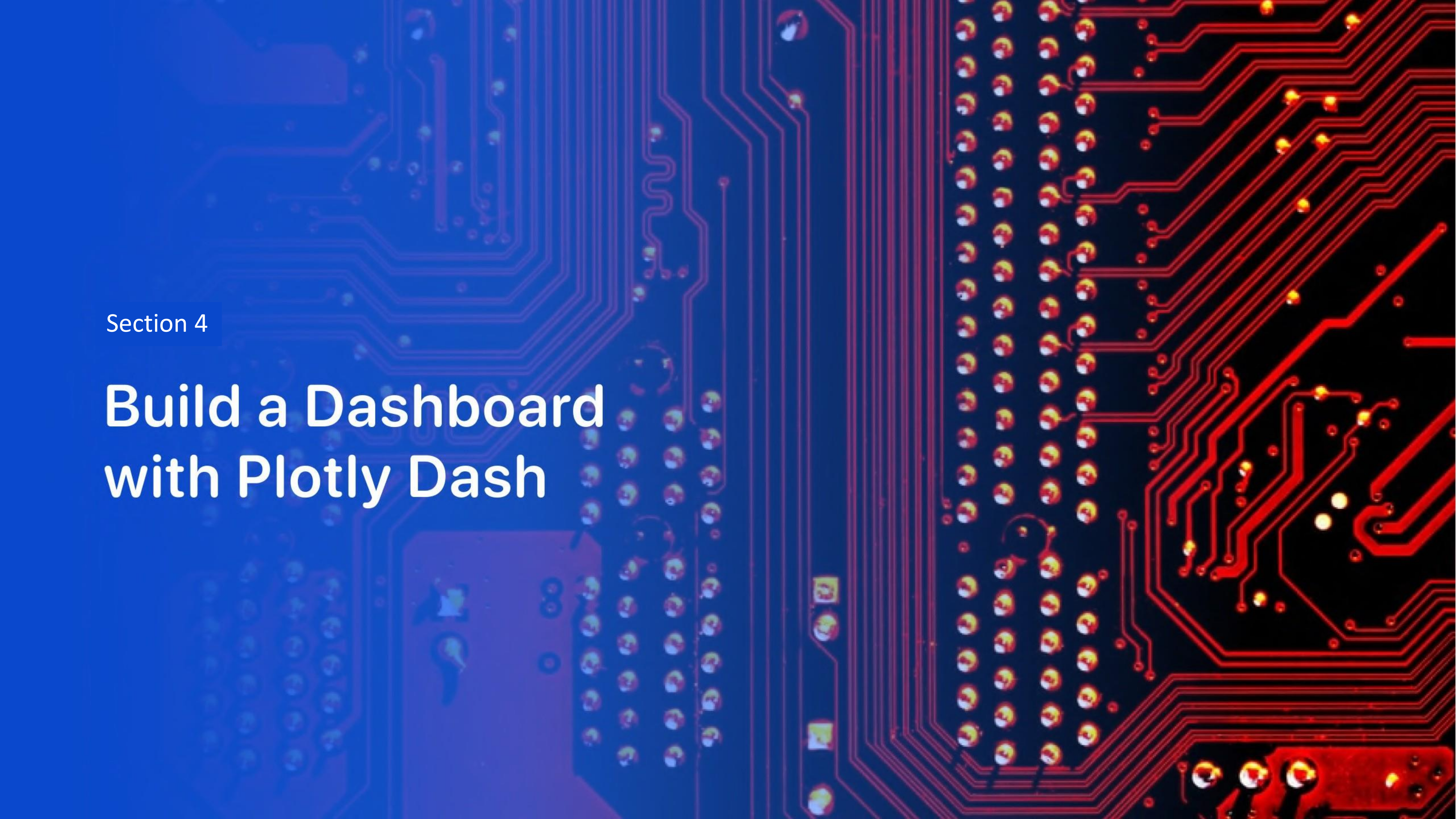








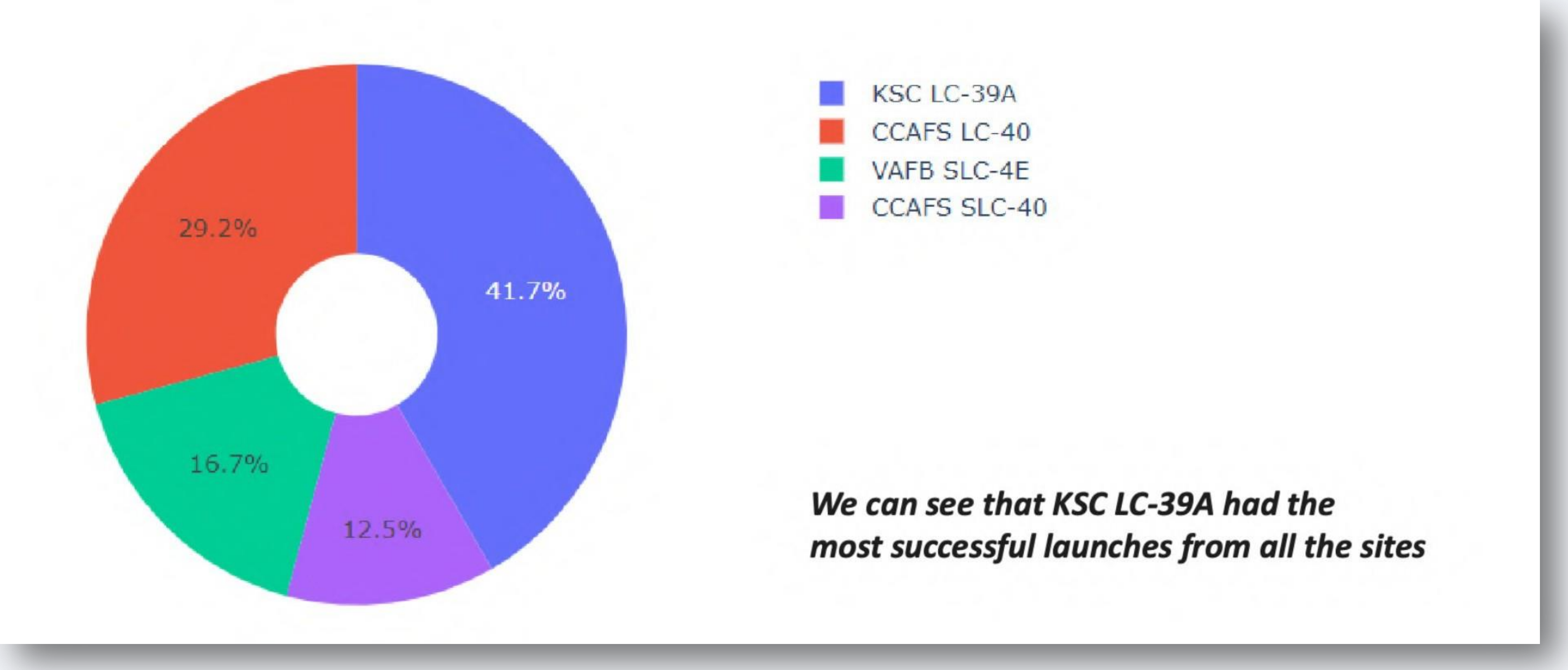
- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes



Section 4

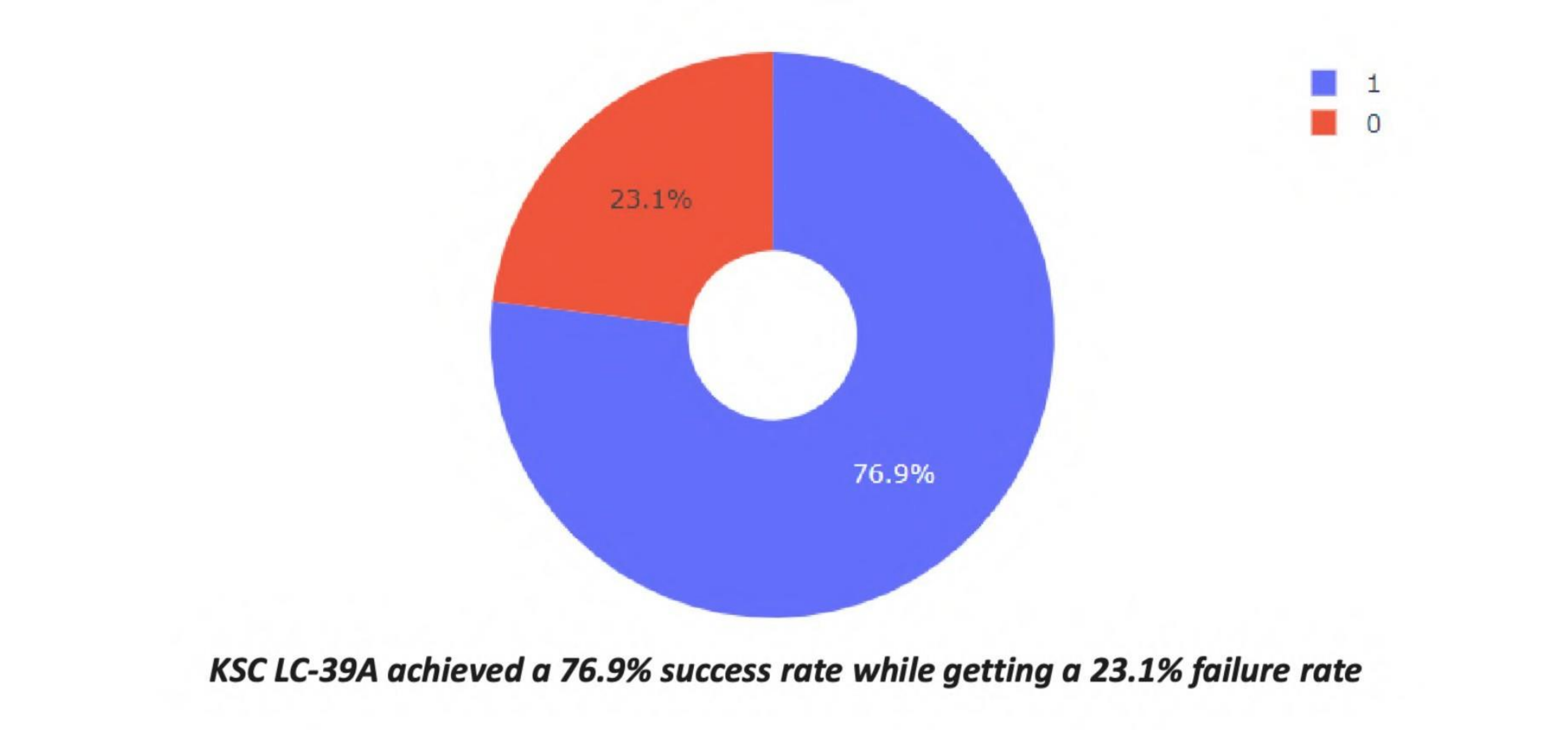
Build a Dashboard with Plotly Dash

The success percentage by each sites.



We can see that KSC LC-39A had the most successful launches from all the sites

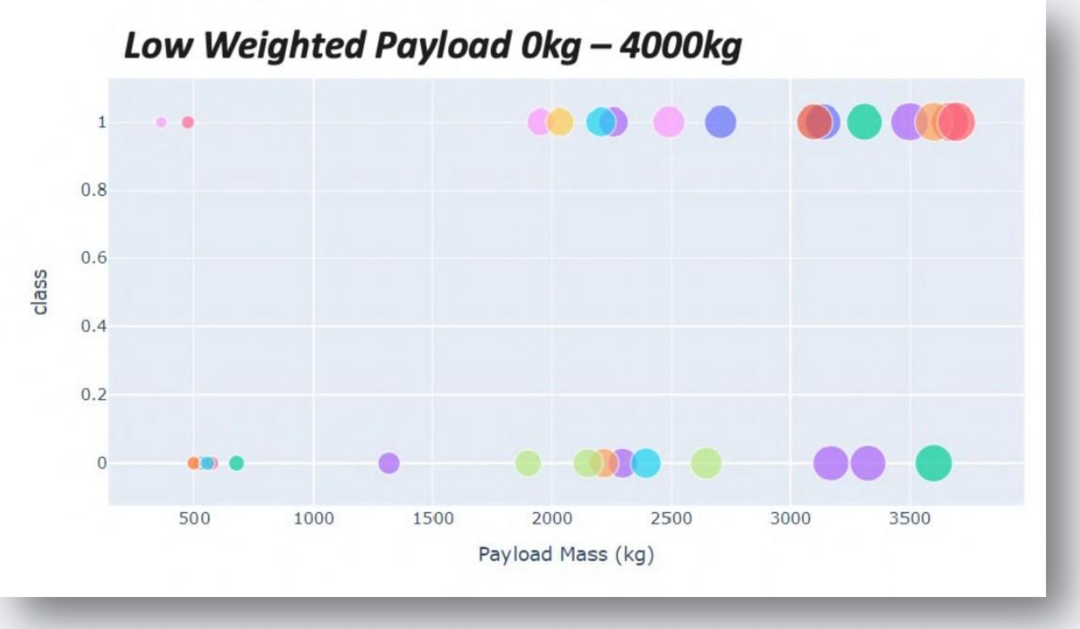
The highest launch-success ratio: KSC LC-39A

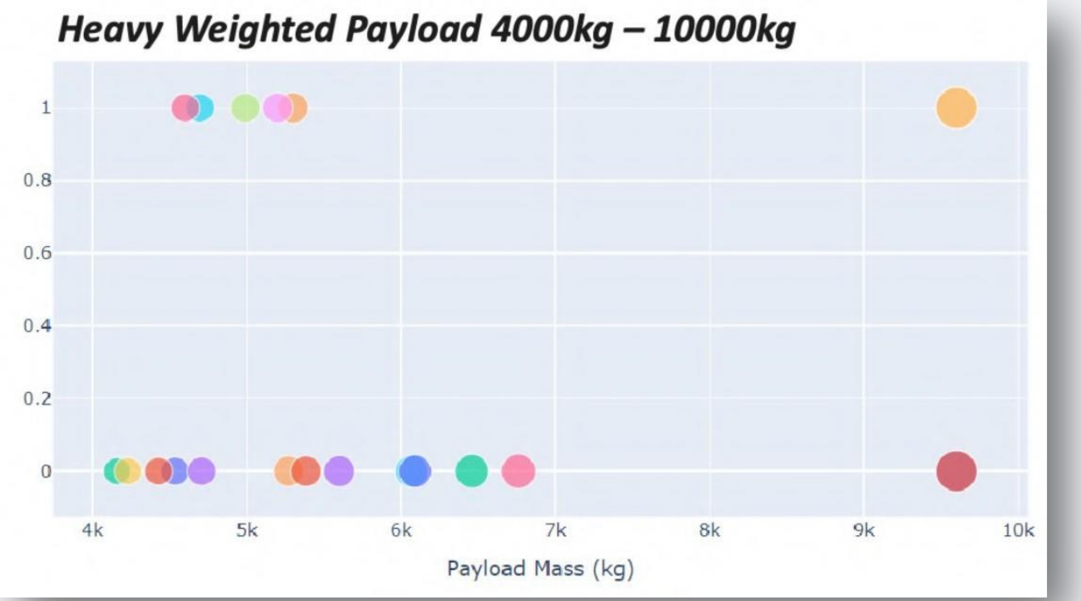


KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

Payload vs Launch Outcome Scatter Plot

We can see that all the success rate for low weighted payload is higher than heavy weighted payload

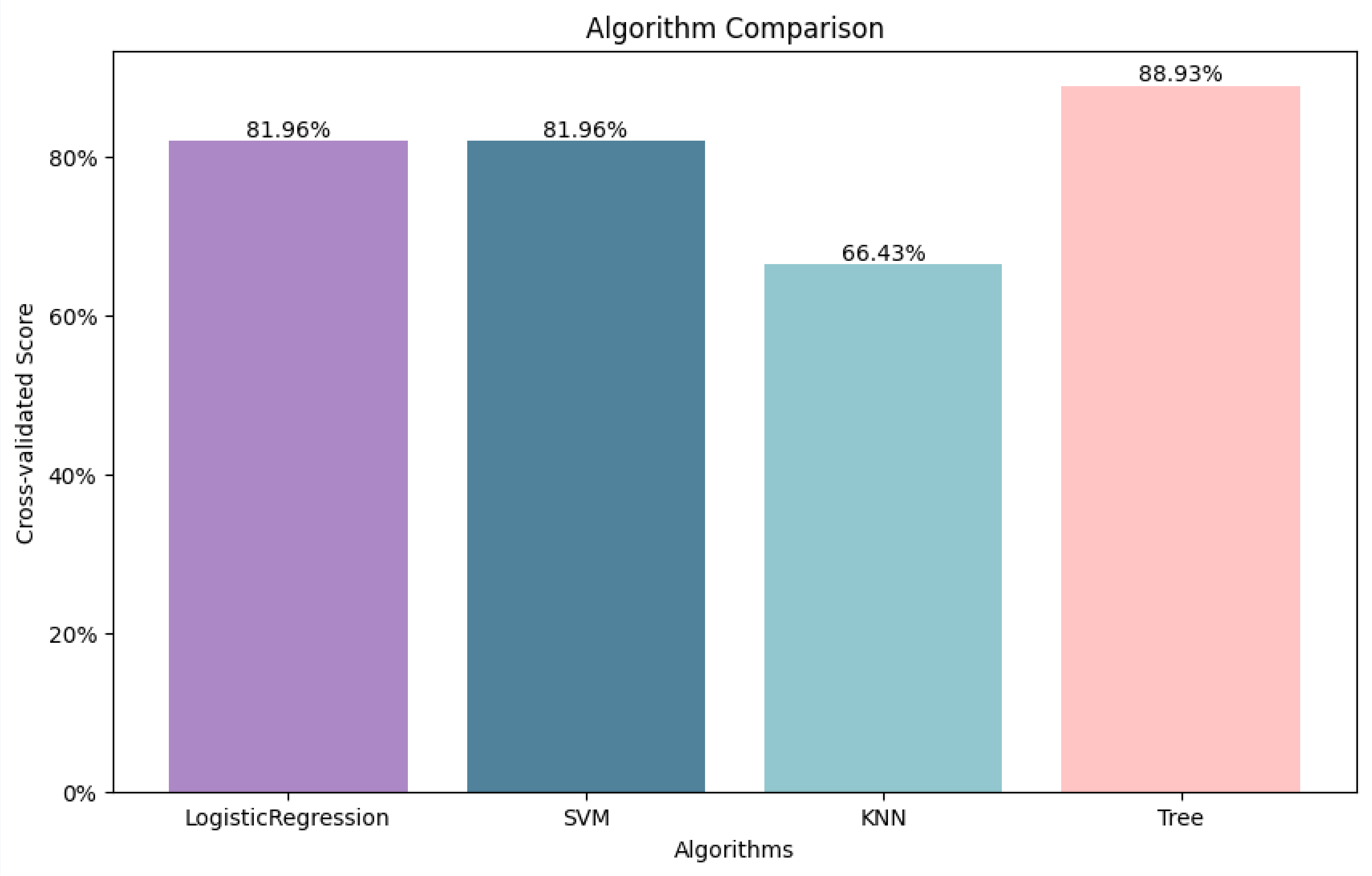




Section 5

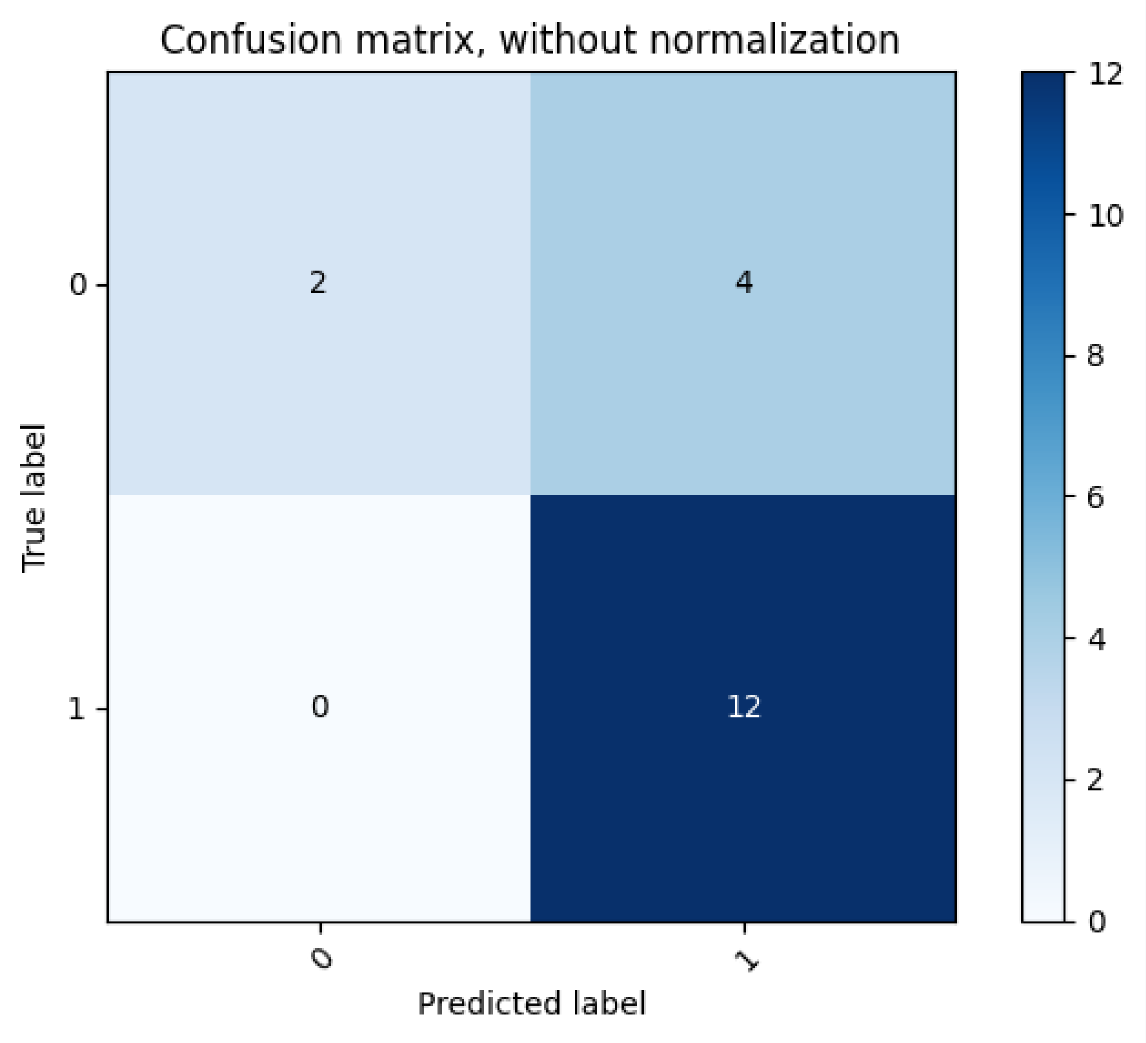
Predictive Analysis (Classification)

Classification Accuracy



As we can see, by using the code as below: we could identify that the best algorithm to be the Tree Algorithm which have the highest classification accuracy.

Confusion Matrix



The confusion matrix for the decision tree classifier shows that the classifier can distinguish between the different classes.
The major problem is the false positives .i.e., unsuccessful landing marked as successful landing by the classifier.

The algorithm was able to predict 14 out of 18 correctly, while 4 predictions were wrong

Conclusions

We can conclude that:

1.Based on our analysis, the Tree Classifier Algorithm emerges as the optimal Machine Learning approach for the given dataset.

2.The performance of payloads weighing 4000kg and below surpassed that of heavier payloads in the dataset.

3.Since 2013, SpaceX has experienced a consistent increase in launch success rates, indicating a trend likely to continue until 2020 and beyond, ultimately leading to enhanced launch precision.

4.KSC LC-39A stands out with the highest success rate among all launch sites at 76.9%.

5.The SSO orbit exhibits the highest success rate of 100% and has occurred more than once.

Thank you!