IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Aneta Rafajová
22/02/2024

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 results
  - Interactive analytics (screenshots)
  - Predictive Analytics results

Introduction

- Project background and context

  SpaceX offers Falcon 9 rocket launches for 62 million dollars, far below competitors' rates of over 165 million dollars. This cost advantage stems from SpaceX's ability to reuse the first stage. Predicting its successful landing is crucial for estimating launch costs and competing bids. The project aims to build a machine learning pipeline for this prediction.

- Problems I want to answer
  - What influences the successful landing of a rocket?
  - How do various features interact to determine landing success?
  - What operational prerequisites are necessary for a successful landing program?

Section 1

Methodology

Methodology

Executive Summary

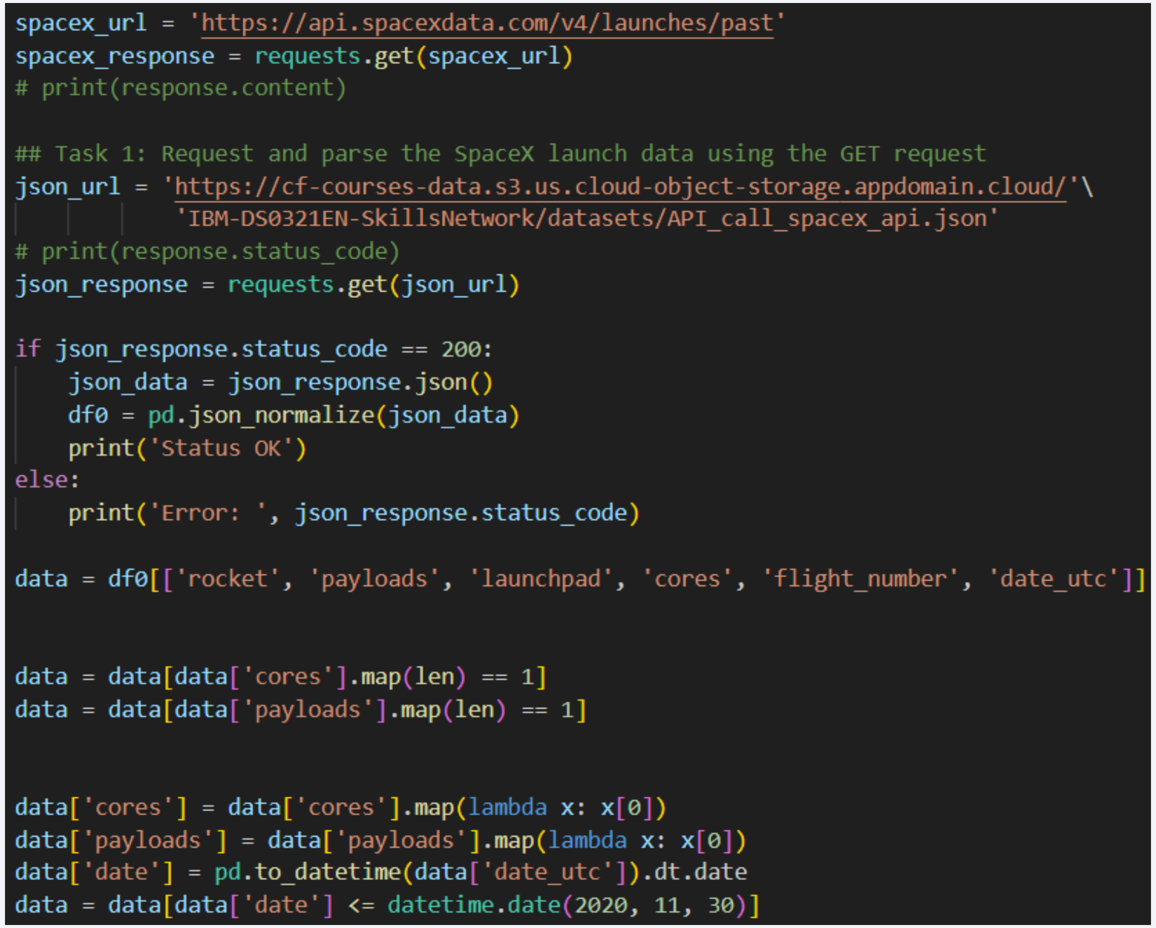
- Data collection methodology:
  - Data was collected with SpaceX API and web scraping
- Perform data wrangling
  - One-hot encoding was applied
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Building, Tuning and evaluating classification models

Data Collection

- Data collection involved making a GET request to the SpaceX API
- Subsequently, we decoded the response content as JSON using the .json() function and transformed it into a pandas dataframe using .json_normalize()
- We proceeded to clean the data, check for missing values, and fill them in where necessary
- Additionally, we conducted web scraping from Wikipedia for Falcon 9 launch records using BeautifulSoup
- The primary objective was to extract the launch records presented as an HTML table, parse the table, and convert it into a pandas dataframe for future analysis

Data Collection – SpaceX API

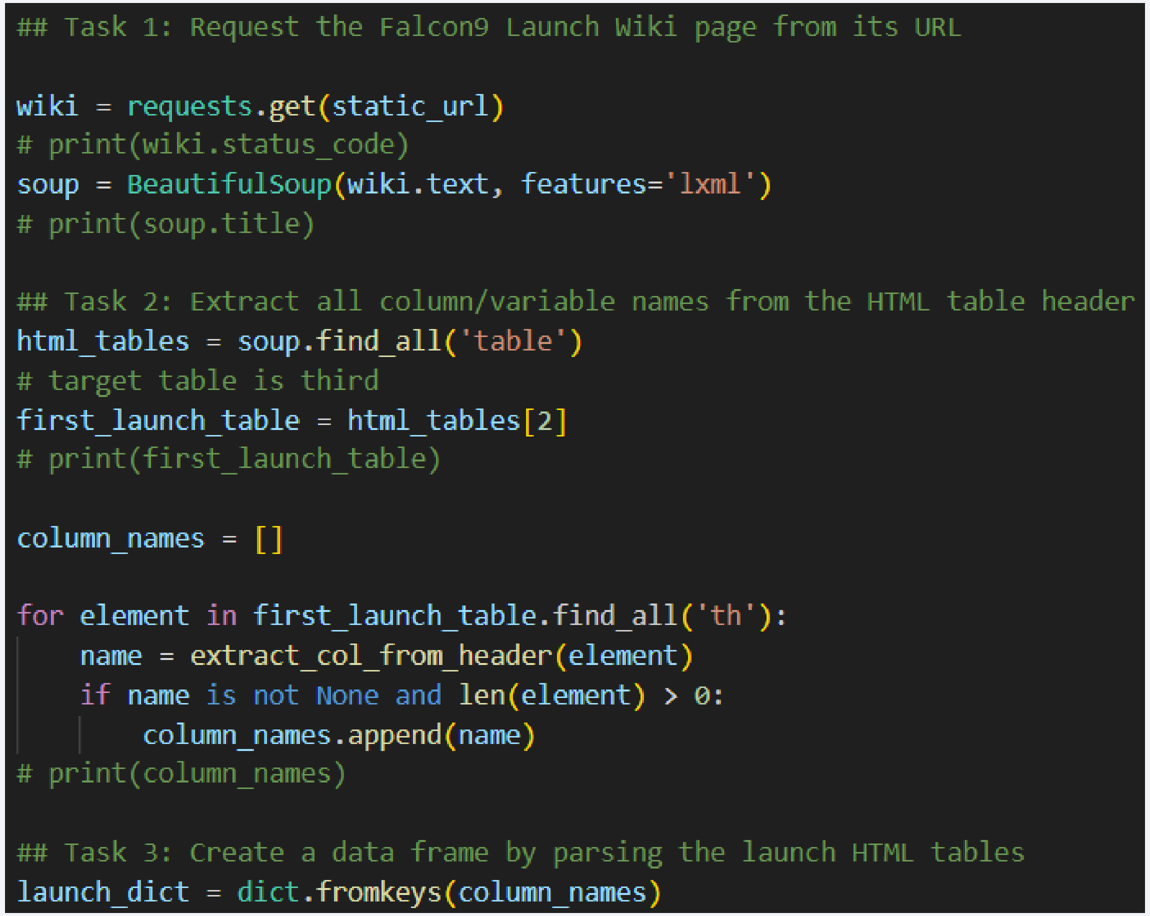
- We utilized a GET request to gather data from the SpaceX API, followed by cleaning and basic data wrangling to refine the collected data and ensure proper formatting
- Link to GitHub: https://github.com/Anet-R/capstone-project/blob/main/01_SpaceX_Data_Collection_API.ipynb



```
spacex_url = 'https://api.spacexdata.com/v4/launches/past'
spacex_response = requests.get(spacex_url)
# print(response.content)

## Task 1: Request and parse the SpaceX launch data using the GET request
json_url = 'https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/'\
            'IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_api.json'
# print(response.status_code)
json_response = requests.get(json_url)

if json_response.status_code == 200:
    json_data = json_response.json()
    df0 = pd.json_normalize(json_data)
    print('Status OK')
else:
    print('Error: ', json_response.status_code)

data = df0[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc']]


data = data[data['cores'].map(len) == 1]
data = data[data['payloads'].map(len) == 1]


data['cores'] = data['cores'].map(lambda x: x[0])
data['payloads'] = data['payloads'].map(lambda x: x[0])
data['date'] = pd.to_datetime(data['date_utc']).dt.date
data = data[data['date'] <= datetime.date(2020, 11, 30)]
```

Data Collection - Scraping

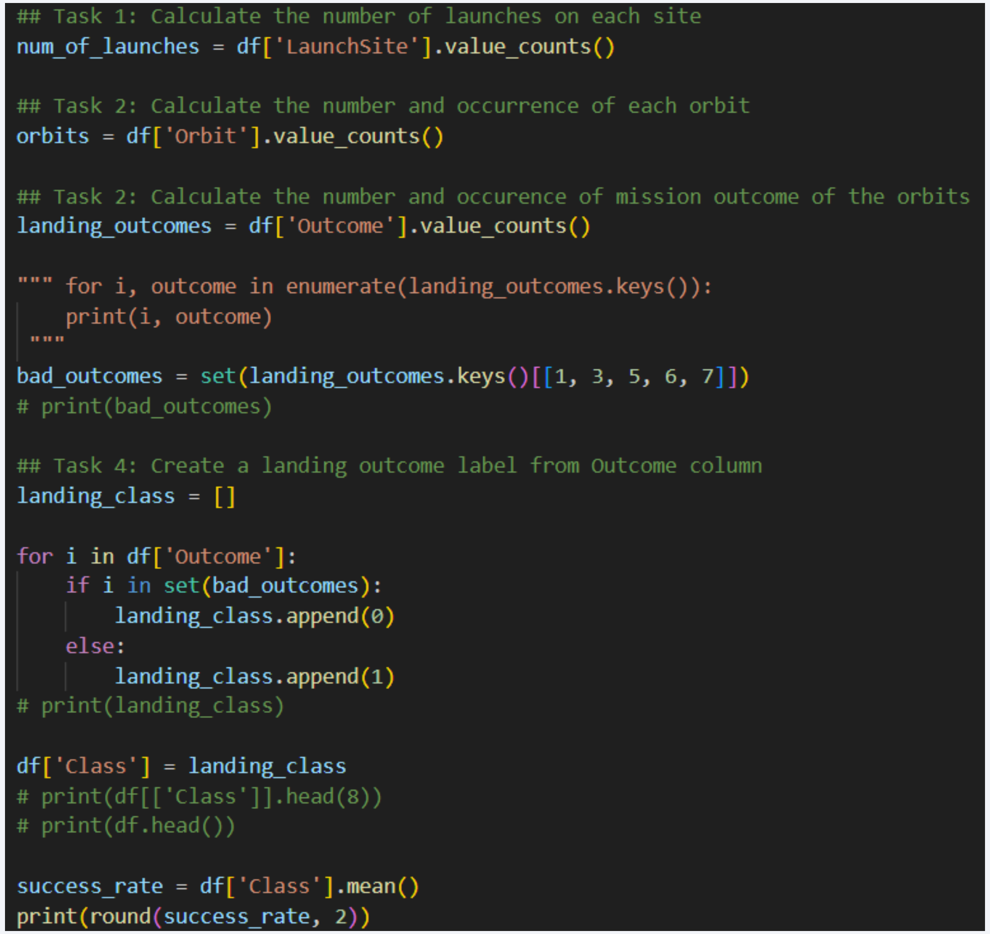
- We employed web scraping techniques with BeautifulSoup to extract Falcon 9 launch records. After parsing the table, we converted the extracted data into a pandas dataframe
- Link to GitHub: https://github.com/Anet-R/capstone-project/blob/main/02_SpaceX_Data_Collection_Web_Scraping.ipynb



```
## Task 1: Request the Falcon9 Launch Wiki page from its URL

wiki = requests.get(static_url)
# print(wiki.status_code)
soup = BeautifulSoup(wiki.text, features='lxml')
# print(soup.title)

## Task 2: Extract all column/variable names from the HTML table header
html_tables = soup.find_all('table')
# target table is third
first_launch_table = html_tables[2]
# print(first_launch_table)

column_names = []

for element in first_launch_table.find_all('th'):
    name = extract_col_from_header(element)
    if name is not None and len(element) > 0:
        column_names.append(name)
# print(column_names)

## Task 3: Create a data frame by parsing the launch HTML tables
launch_dict = dict.fromkeys(column_names)
```

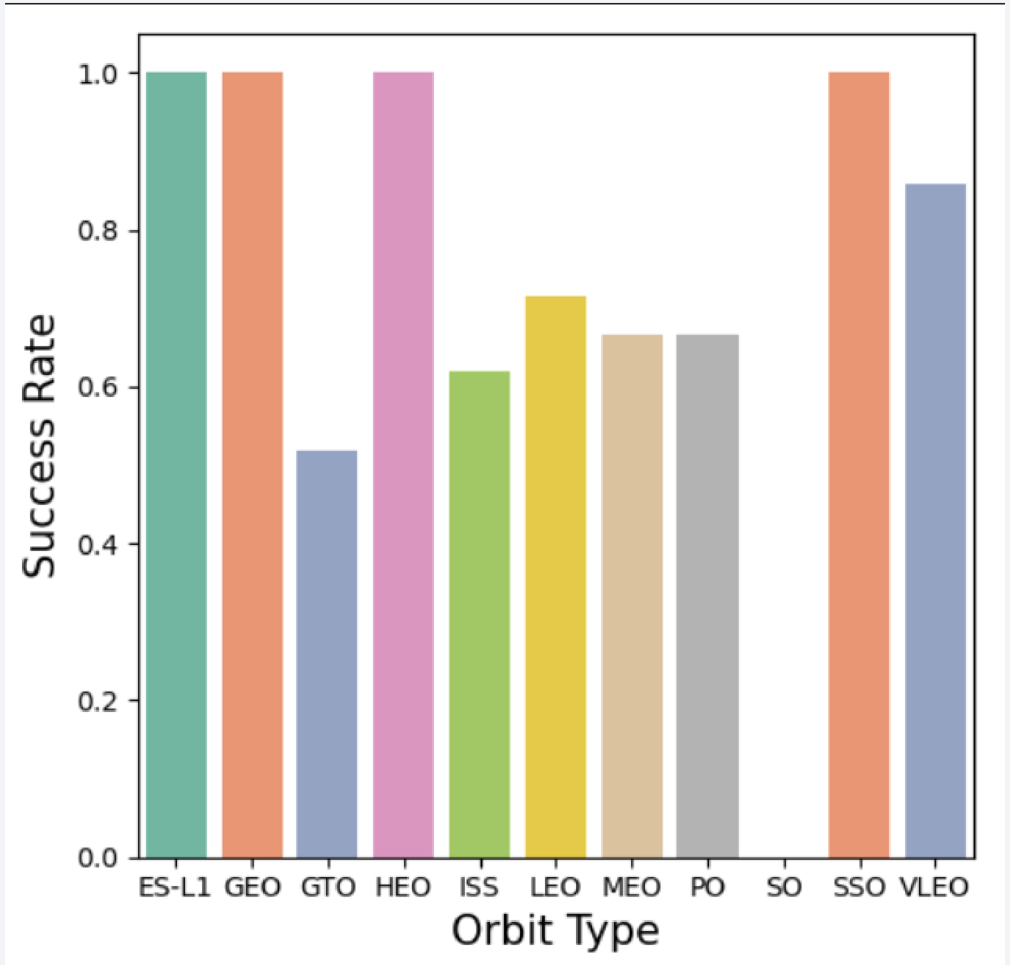
Data Wrangling

- Through exploratory data analysis, we identified and defined the training labels.
- We computed the frequency of launches at each site and the occurrence of each orbit.
- From the outcome column, we derived a landing outcome label and exported the results to a CSV file
- Link to GitHub: https://github.com/Anet-R/capstone-project/blob/main/03_SpaceX_Data_Wrangling.ipynb



```
## Task 1: Calculate the number of launches on each site
num_of_launches = df['LaunchSite'].value_counts()

## Task 2: Calculate the number and occurrence of each orbit
orbits = df['Orbit'].value_counts()

## Task 2: Calculate the number and occurence of mission outcome of the orbits
landing_outcomes = df['Outcome'].value_counts()

""" for i, outcome in enumerate(landing_outcomes.keys()):
    print(i, outcome)
 """
bad_outcomes = set(landing_outcomes.keys()[[1, 3, 5, 6, 7]])
# print(bad_outcomes)

## Task 4: Create a landing outcome label from Outcome column
landing_class = []

for i in df['Outcome']:
    if i in set(bad_outcomes):
        landing_class.append(0)
    else:
        landing_class.append(1)
# print(landing_class)

df['Class'] = landing_class
# print(df[['Class']].head(8))
# print(df.head())

success_rate = df['Class'].mean()
print(round(success_rate, 2))
```

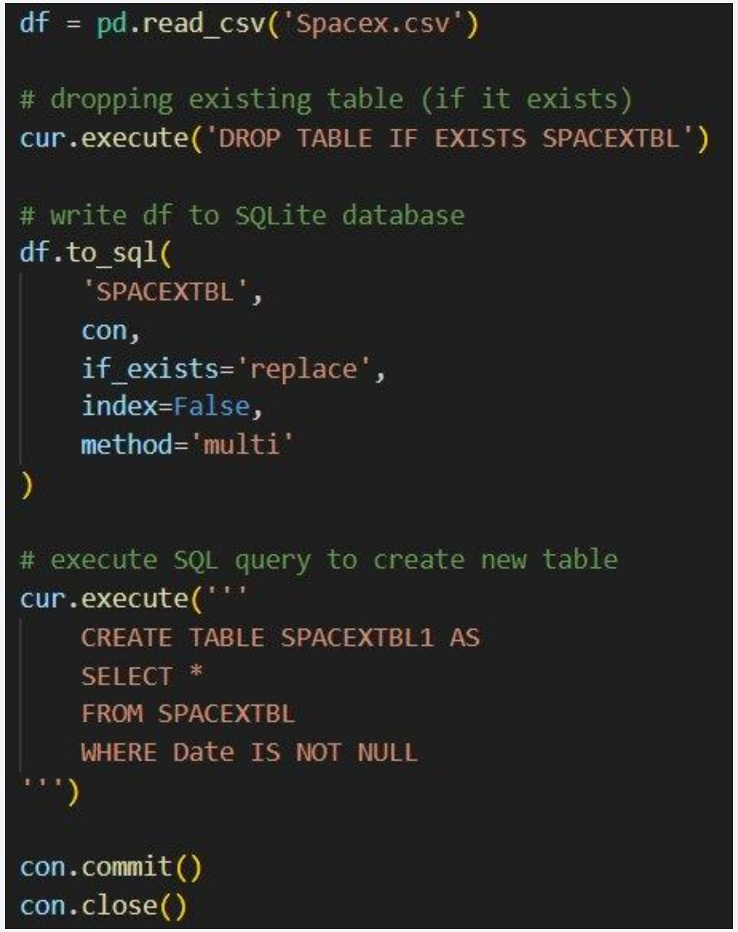
EDA with Data Visualization



- We conducted data exploration by visualizing various relationships, including:
    - The correlation between flight number and launch site
    - The relationship between payload and launch site
    - The success rate of each orbit type
    - The connection between flight number and orbit type
    - The yearly trend in launch success
- To GitHub: https://github.com/Anet-R/capstone-project/blob/main/05_SpaceX_EDA_Data_Visualization.ipynb

EDA with SQL

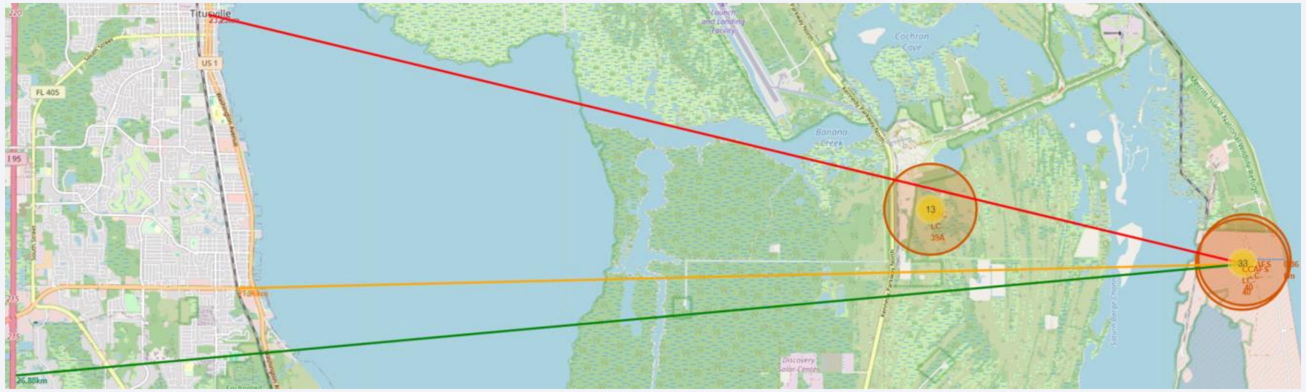
- We seamlessly loaded the SpaceX dataset into a PostgreSQL database directly from the Jupyter Notebook environment
- Employing SQL for exploratory data analysis, we derived insights from the dataset by crafting queries to investigate various aspects, such as:
  - The names of unique launch sites in 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.
- Link to GitHub: https://github.com/Anet-R/capstone-project/blob/main/04_SpaceX_EDA_SQL.ipynb



```
df = pd.read_csv('Spacex.csv')

# dropping existing table (if it exists)
cur.execute('DROP TABLE IF EXISTS SPACEXTBL')

# write df to SQLite database
df.to_sql(
    'SPACEXTBL',
    con,
    if_exists='replace',
    index=False,
    method='multi'
)

# execute SQL query to create new table
cur.execute('''
    CREATE TABLE SPACEXTBL1 AS
    SELECT *
    FROM SPACEXTBL
    WHERE Date IS NOT NULL
''')

con.commit()
con.close()
```

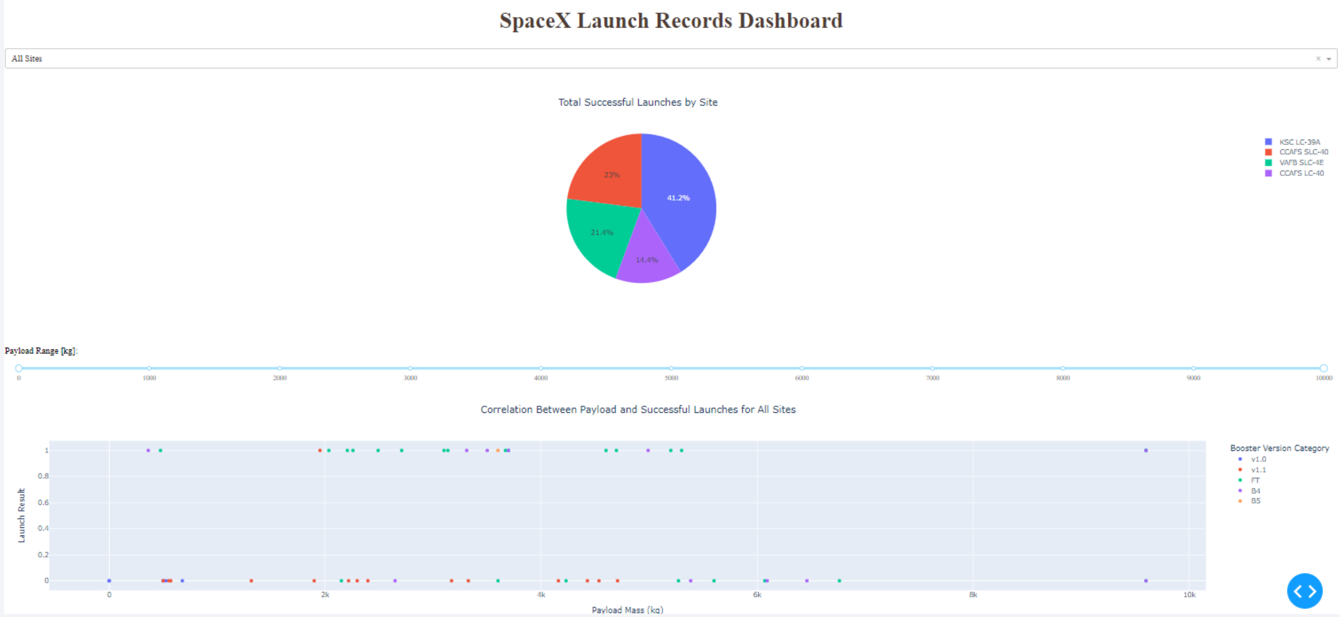
Build an Interactive Map with Folium

- We annotated all launch sites on a Folium map, integrating map objects like markers, circles, and lines to denote launch success or failure for each site
- By assigning launch outcomes (failure or success) to classes 0 and 1, respectively, we facilitated analysis
- Leveraging color-coded marker clusters, we discerned launch sites with notably high success rates
- We computed distances between launch sites and their surroundings, addressing queries such as:
  - We investigated whether launch sites are located near railways, highways, and coastlines
  - We examined whether launch sites maintain a certain distance from cities
- Link to GitHub: https://github.com/Anet-R/capstone-project/blob/main/06_SpaceX_IVA_Folium.ipynb



Build a Dashboard with Plotly Dash

- We developed an interactive dashboard using Plotly Dash.
- We created pie charts illustrating the total launches from specific sites.
- We generated scatter plots to visualize the relationship between outcome and payload mass (in kilograms) for various booster versions.
- Link to GitHub: https://github.com/Anet-R/capstone-project/blob/main/07_SpaceX_IVA_Plotly.py



SpaceX Launch Records Dashboard

All Sites

Total Successful Launches by Site

Payload Range [kg]:

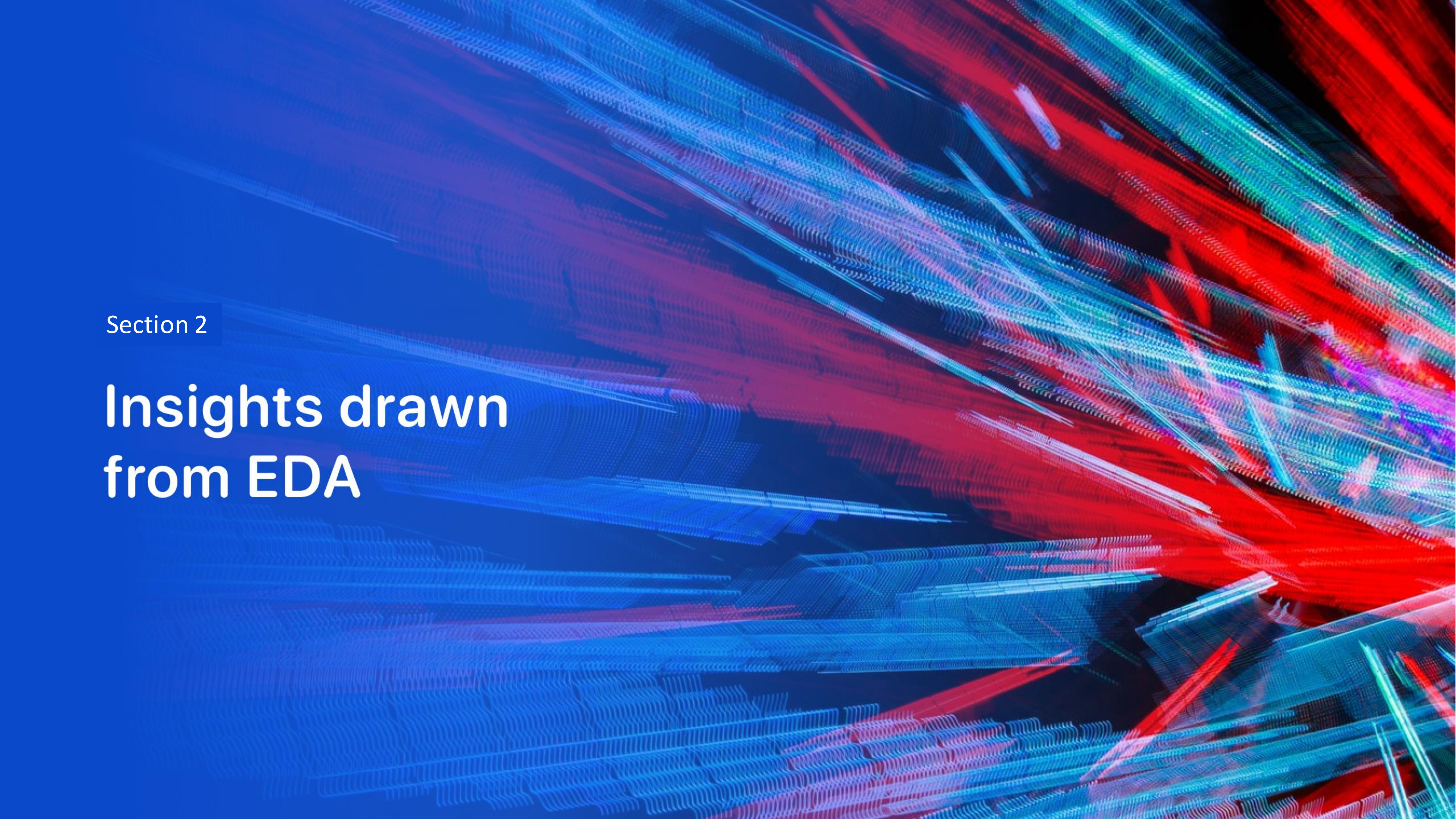
Correlation Between Payload and Successful Launches for All Sites

Predictive Analysis (Classification)

- We utilized NumPy and Pandas to load the data, perform transformations, and split it into training and testing sets.
- Employing GridSearchCV, we built multiple machine learning models and fine-tuned hyperparameters for optimization.
- With accuracy as our chosen metric, we iteratively enhanced the model through feature engineering and algorithm tuning.
- Ultimately, we identified the best-performing classification model from our evaluations.
- Link to GitHub: https://github.com/Anet-R/capstone-project/blob/main/08_SpaceX_Predictive_Analysis.ipynb

Results

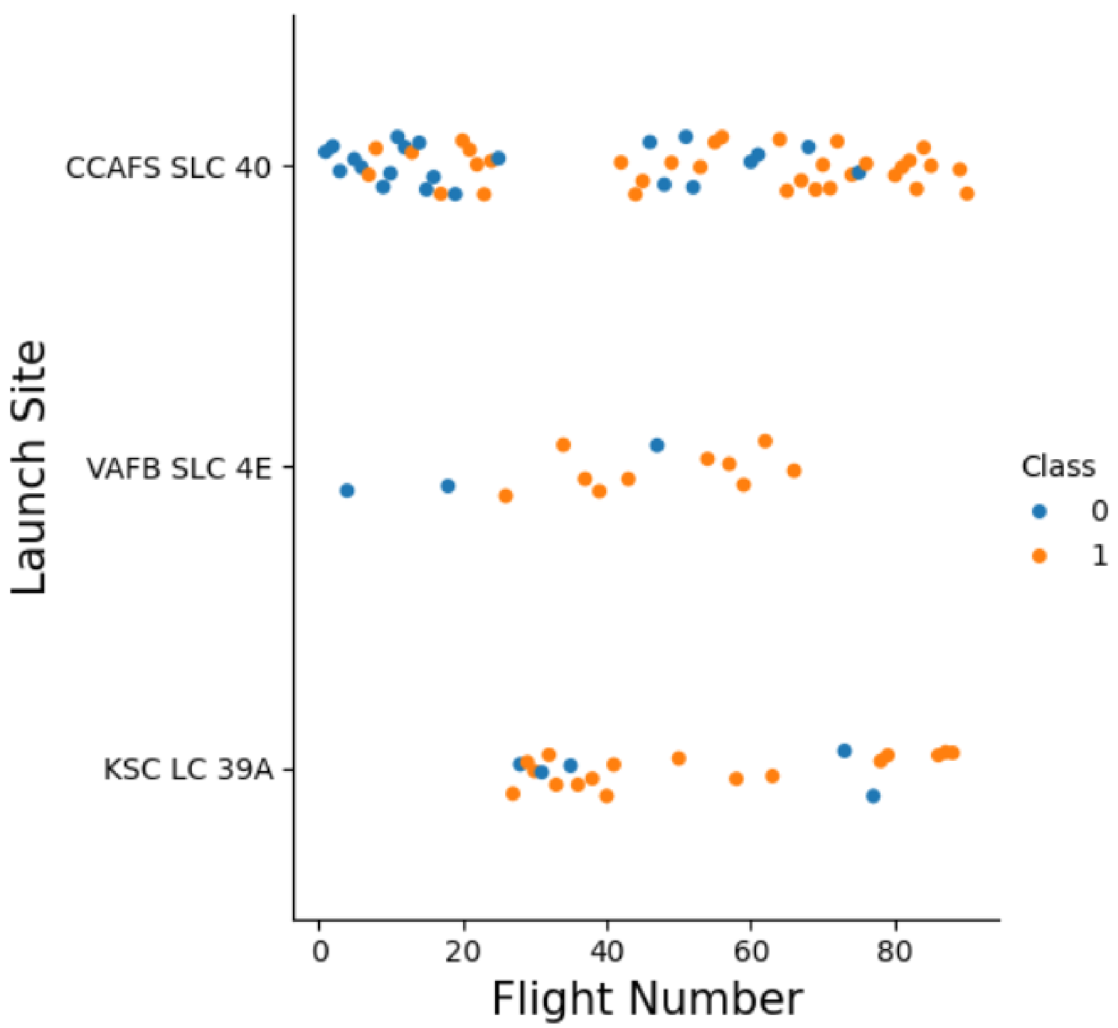
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results



Section 2

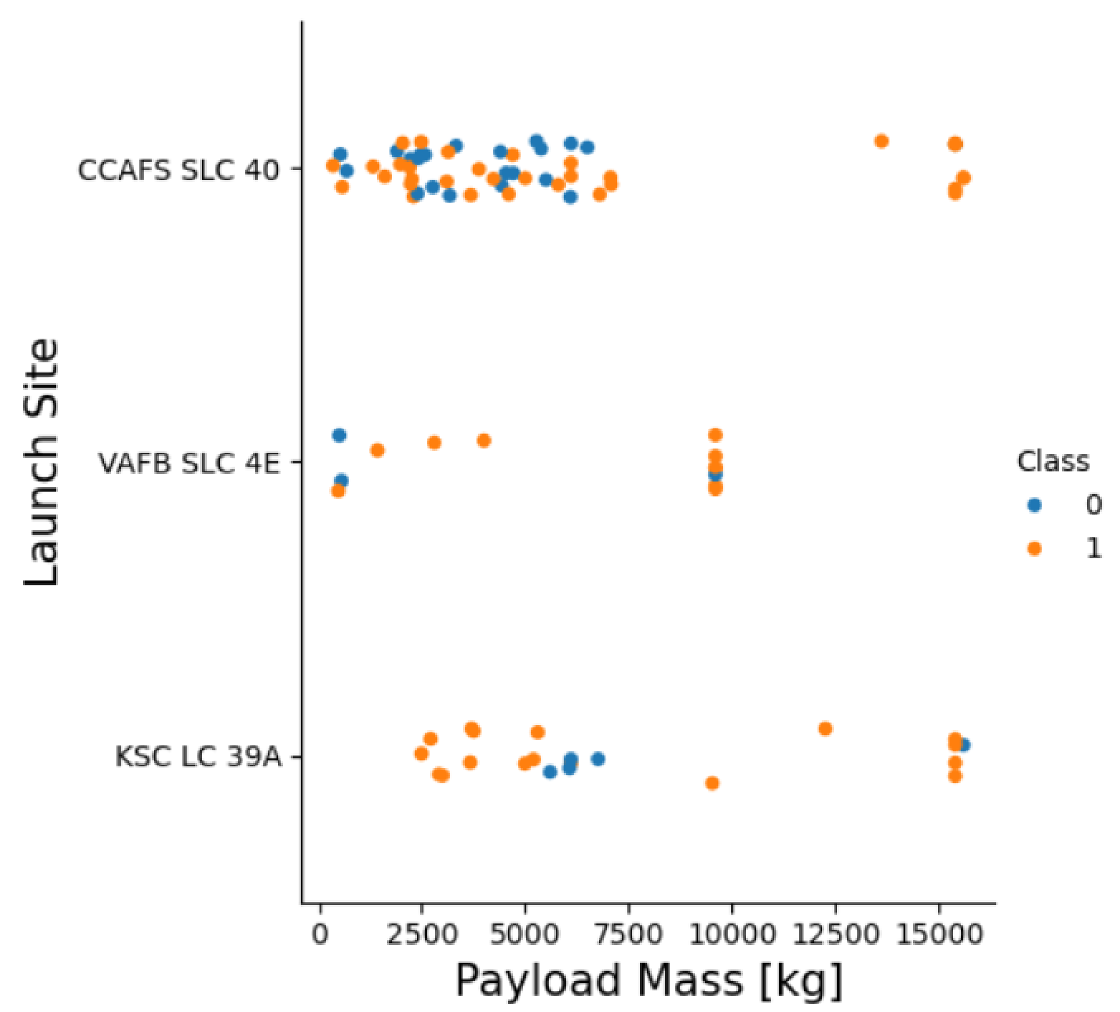
Insights drawn from EDA

Flight Number vs. Launch Site



The plot indicates a direct relationship: higher flight volume at a launch site correlates with a greater success rate

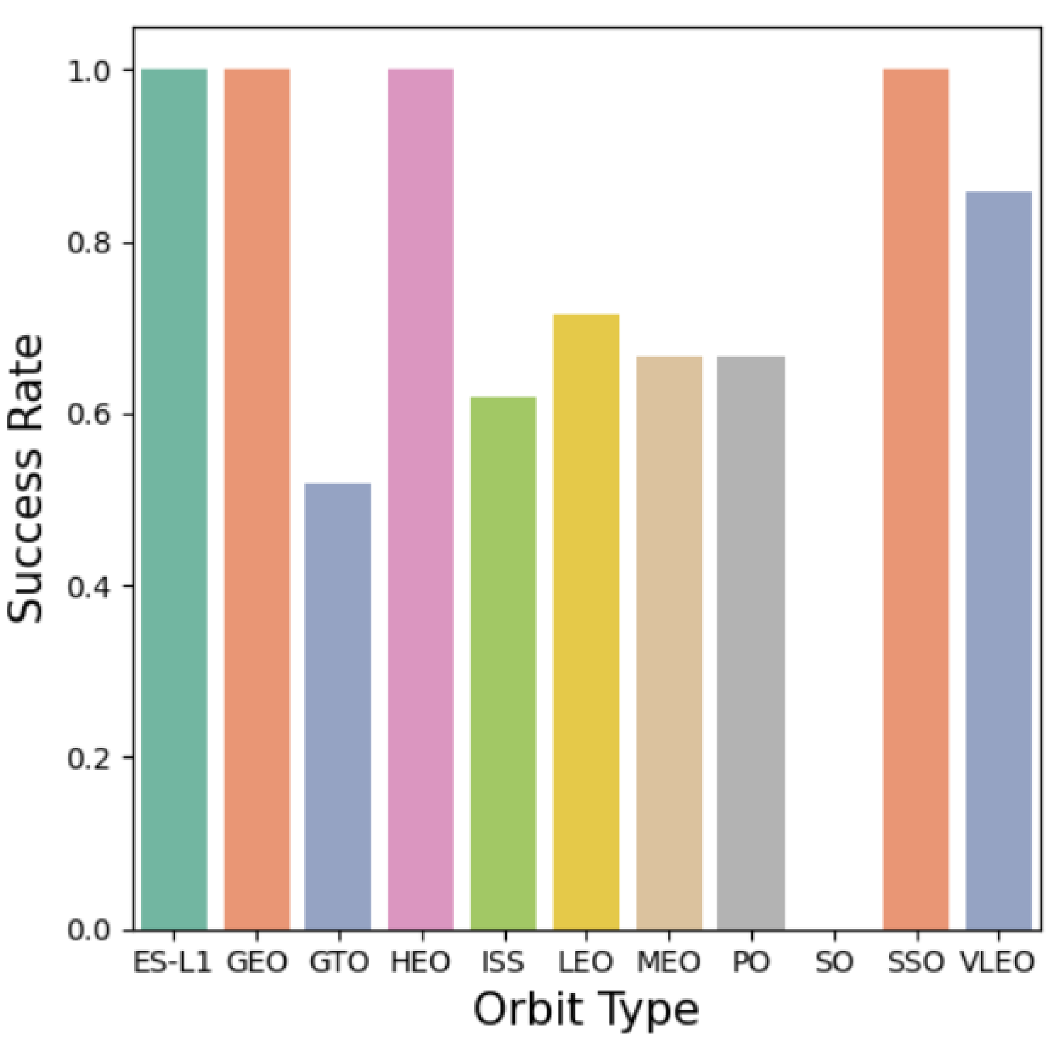
Payload vs. Launch Site



There's a positive correlation between payload mass and rocket success rate at the launch site

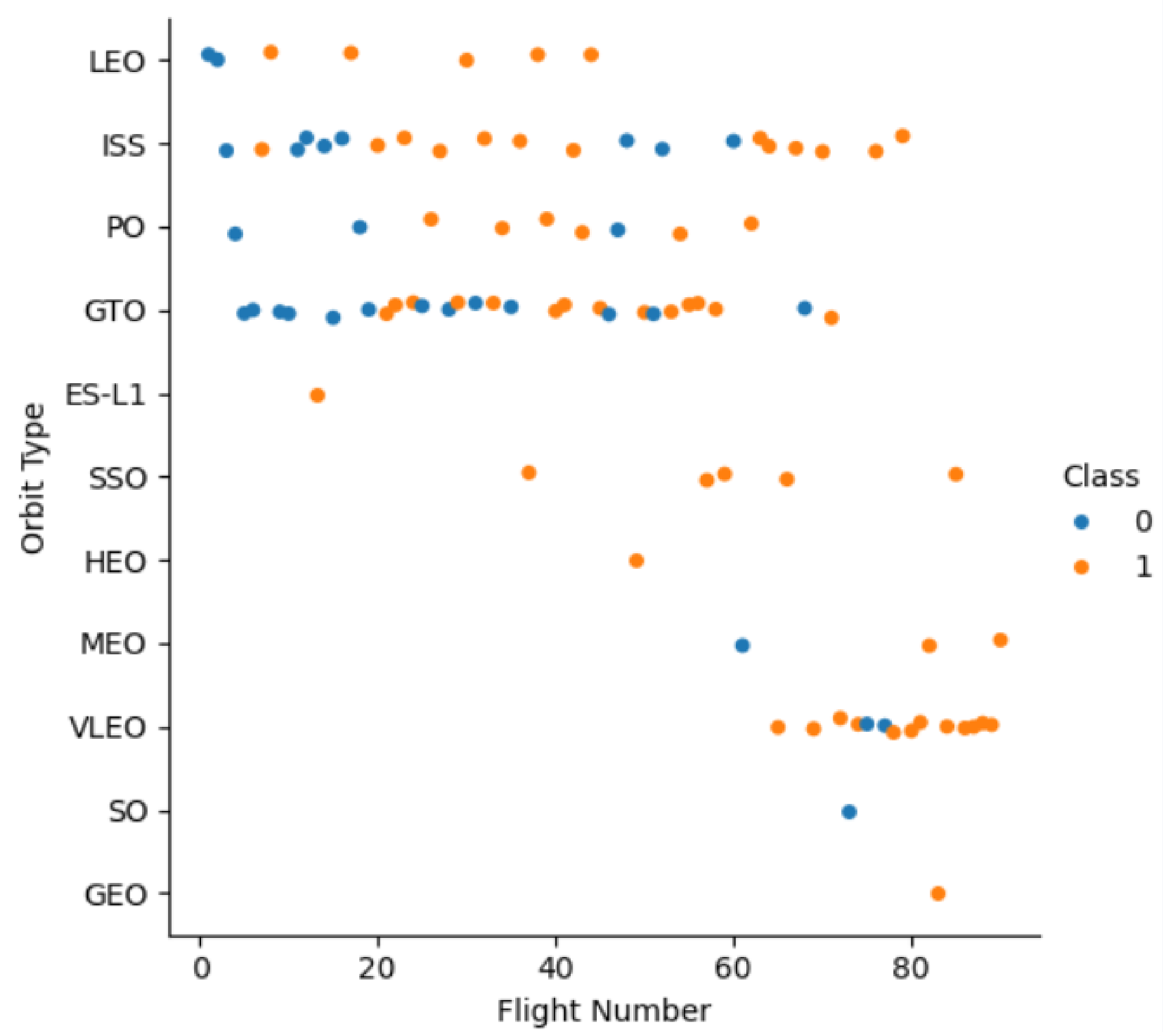
Success Rate vs. Orbit Type

The plot highlights that ES-L1, GEO, HEO, SSO, and VLEO exhibited the highest success rates

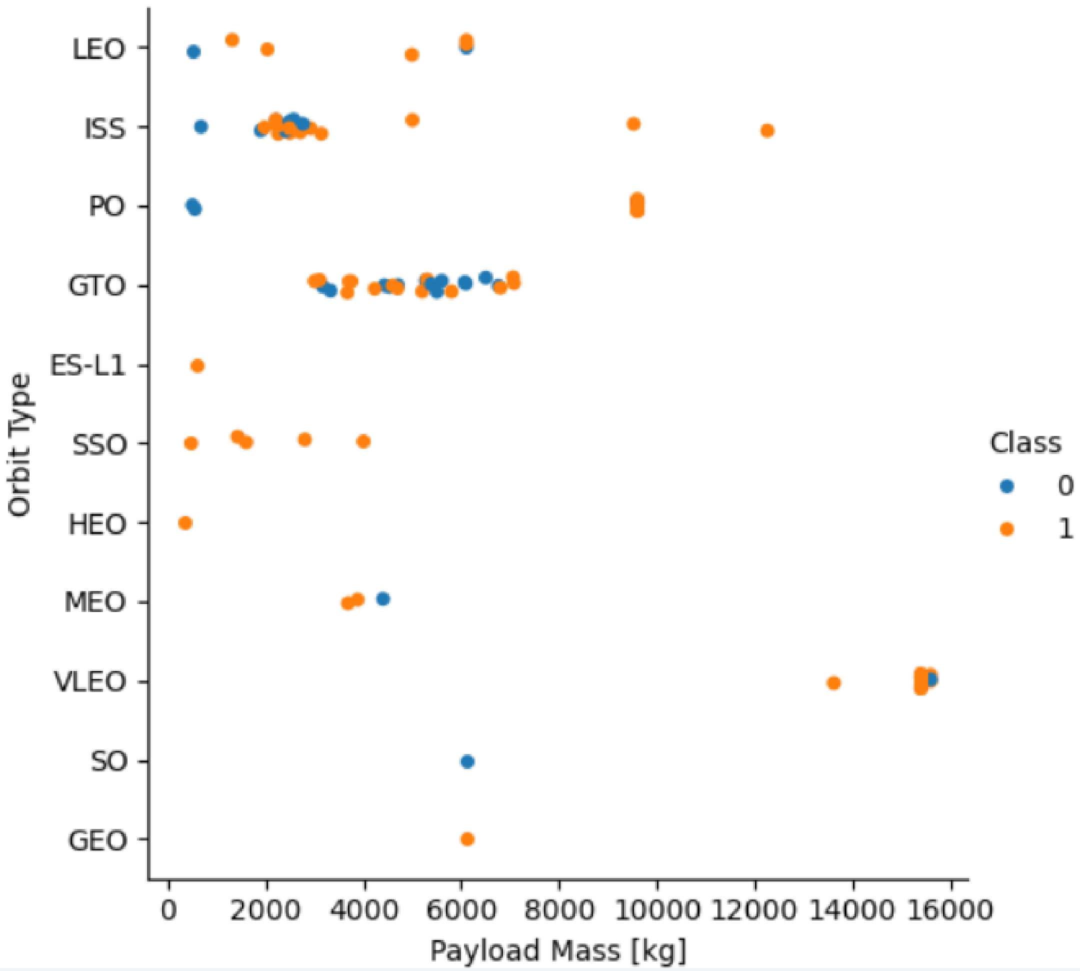


Flight Number vs. Orbit Type

The plot below illustrates the relationship between Flight Number and Orbit type. It is evident that in the LEO orbit, success appears to be influenced by the number of flights, whereas in the GTO orbit, there is no discernible relationship between flight number and orbit success



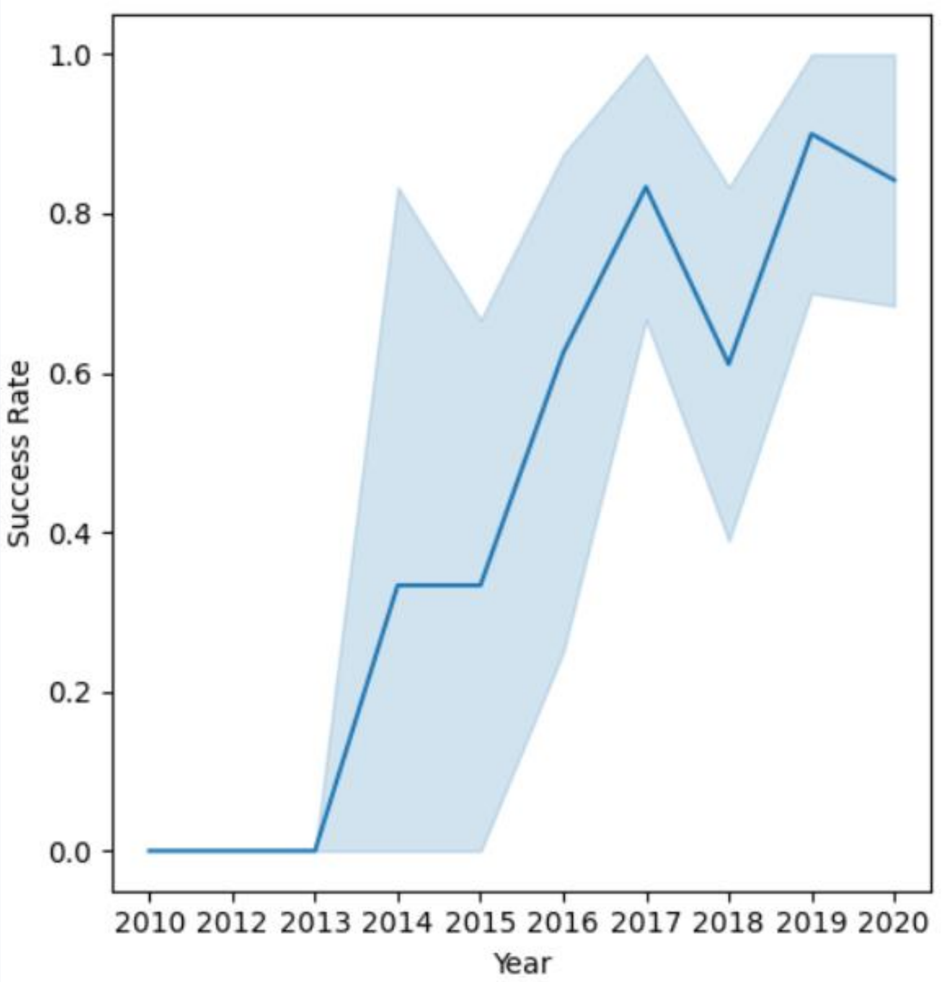
Payload vs. Orbit Type



It's noticeable that heavier payloads correspond to a higher frequency of successful landings, particularly for PO, LEO, and ISS orbits

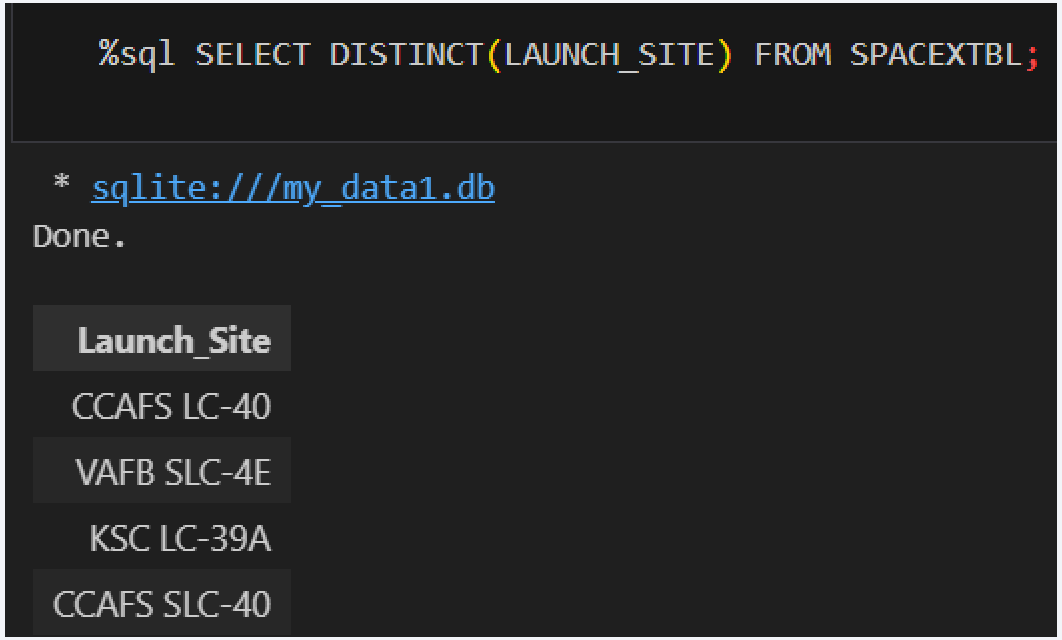
Launch Success Yearly Trend

The plot illustrates that the success rate has steadily increased since 2013, peaking in 2020



All Launch Site Names

We employed the keyword DISTINCT to display only unique launch sites extracted from the SpaceX dataset

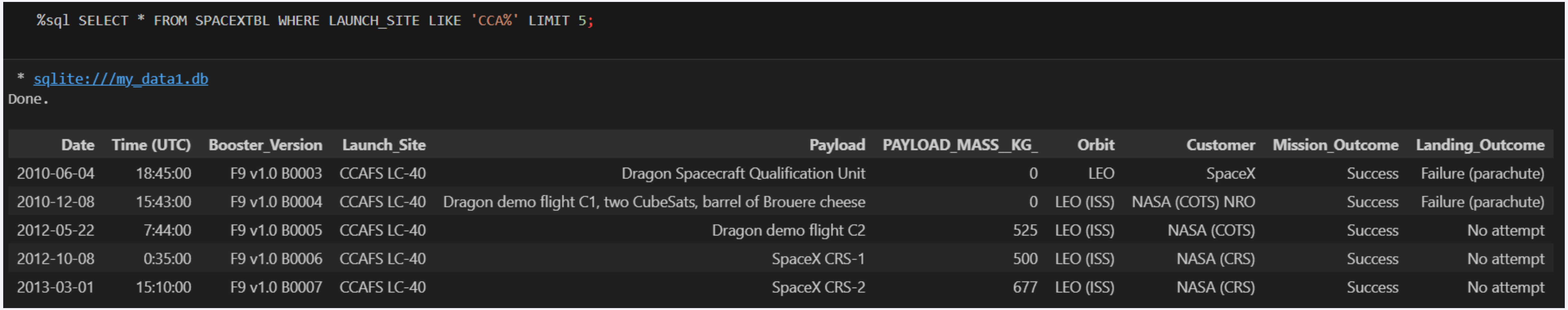


```
%sql SELECT DISTINCT(LAUNCH_SITE) FROM SPACEXTBL;
```

```
 * 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'



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

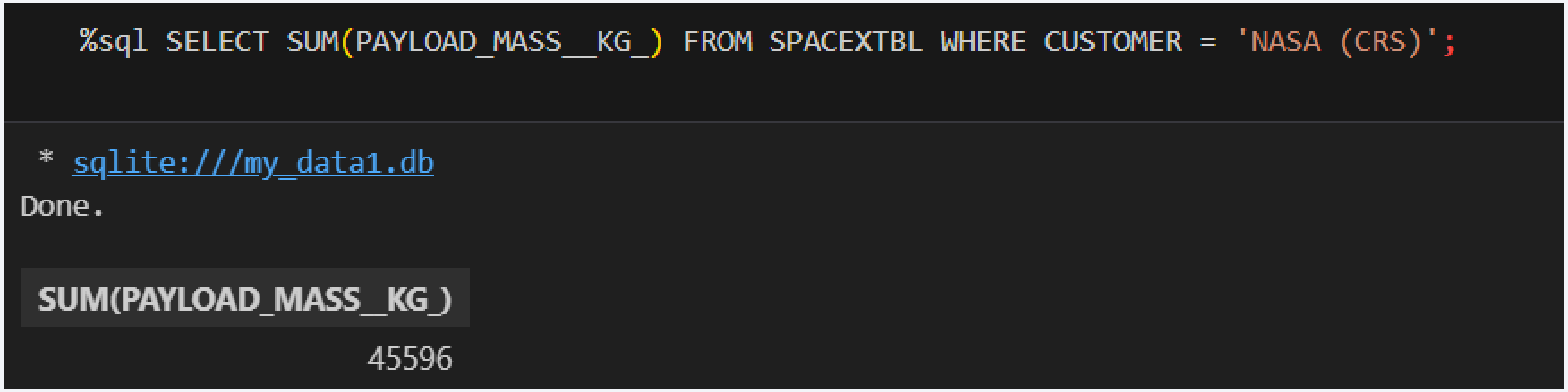
```
 * sqlite:///my_data1.db
Done.
```

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	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)
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

We utilized the aforementioned query to showcase 5 records where launch sites commence with CCA.

Total Payload Mass

Using the query provided below, we computed the total payload carried by boosters from NASA as 45596

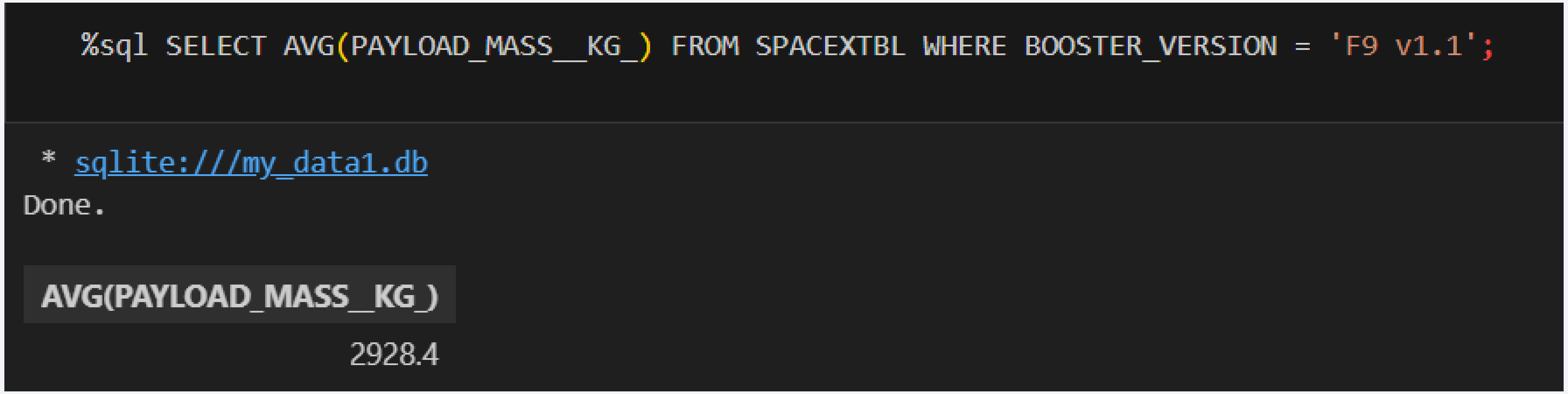


```
%sql SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE CUSTOMER = 'NASA (CRS)';
```

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

SUM(PAYLOAD_MASS__KG_)
                 45596
```

Average Payload Mass by F9 v1.1

The average payload mass carried by booster version F9 v1.1 was calculated to be 2928.4 using the query below

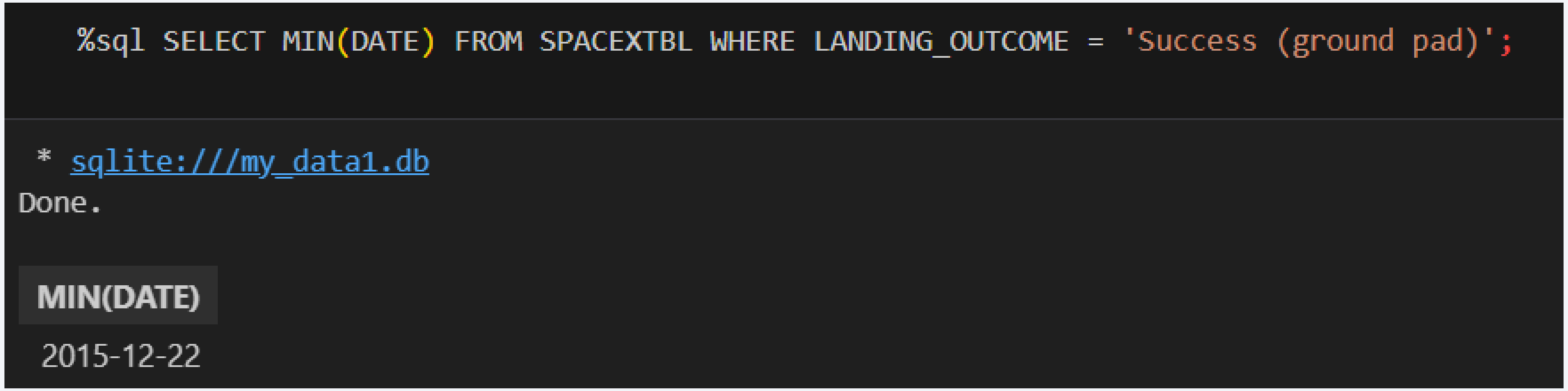


```
%sql SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE BOOSTER_VERSION = 'F9 v1.1';
```

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

AVG(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date



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

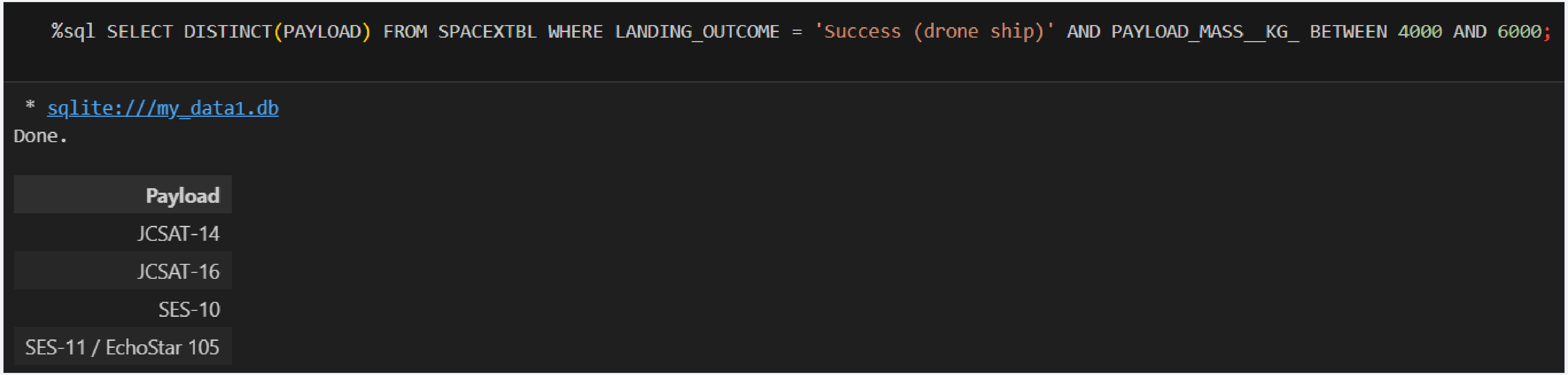
```
 * sqlite:///my_data1.db
Done.
```

MIN(DATE)
2015-12-22

We noted that the date of the first successful landing outcome on the ground pad was December 22nd, 2015

Successful Drone Ship Landing with Payload between 4000 and 6000

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



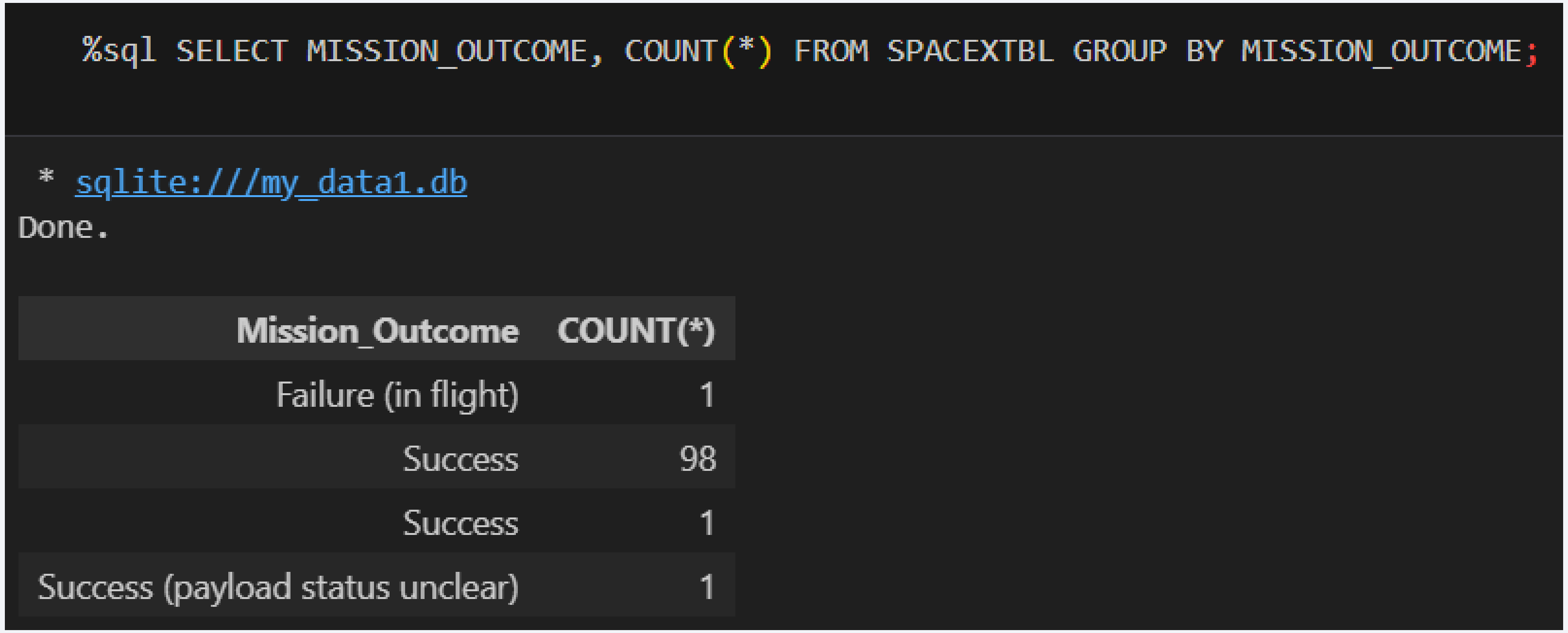
```
%sql SELECT DISTINCT(PAYLOAD) FROM SPACEXTBL WHERE LANDING_OUTCOME = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000;
```

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

Payload
JCSAT-14
JCSAT-16
SES-10
SES-11 / EchoStar 105

Total Number of Successful and Failure Mission Outcomes

We utilized wildcard '%' to filter for WHERE MissionOutcome where the outcome was either a success or a failure

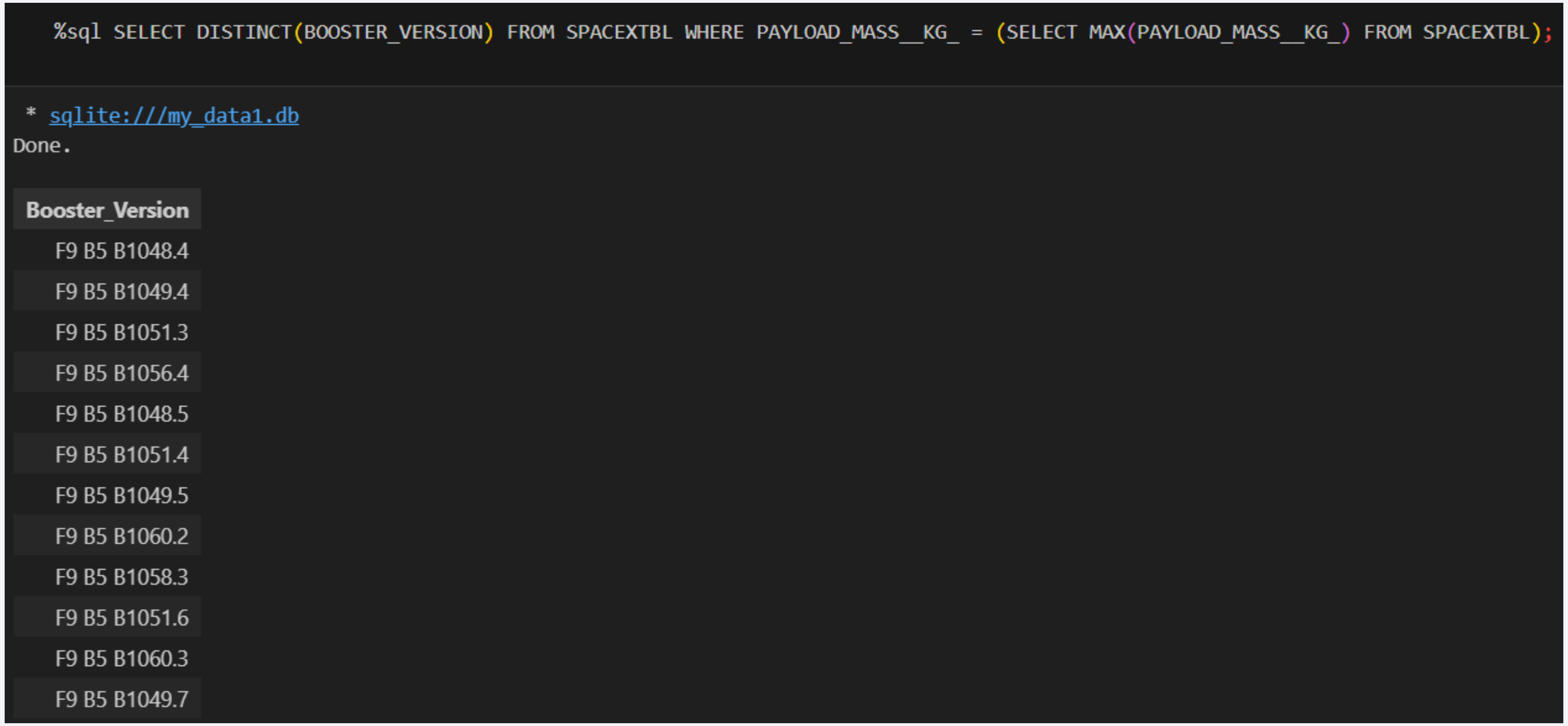


```
%sql SELECT MISSION_OUTCOME, COUNT(*) FROM SPACEXTBL GROUP BY MISSION_OUTCOME;
```

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

Mission_Outcome	COUNT(*)
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Boosters Carried Maximum Payload



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

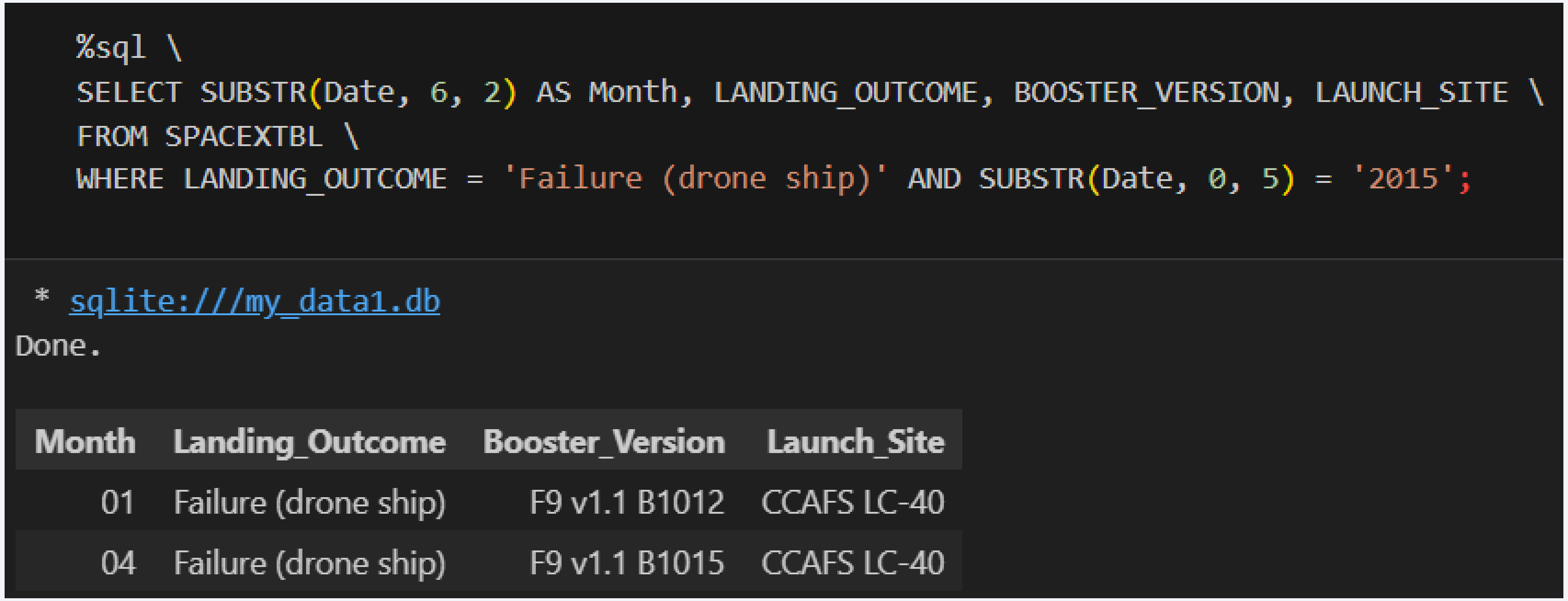
```
 * sqlite:///my_data1.db
Done.
```

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

We identified the booster that has carried the maximum payload by employing a subquery within the WHERE clause alongside the MAX() function

2015 Launch Records

We employed a combination of the WHERE clause, LIKE, AND, and BETWEEN conditions to filter for failed landing outcomes on drone ships, along with their booster versions and launch site names, for the year 2015

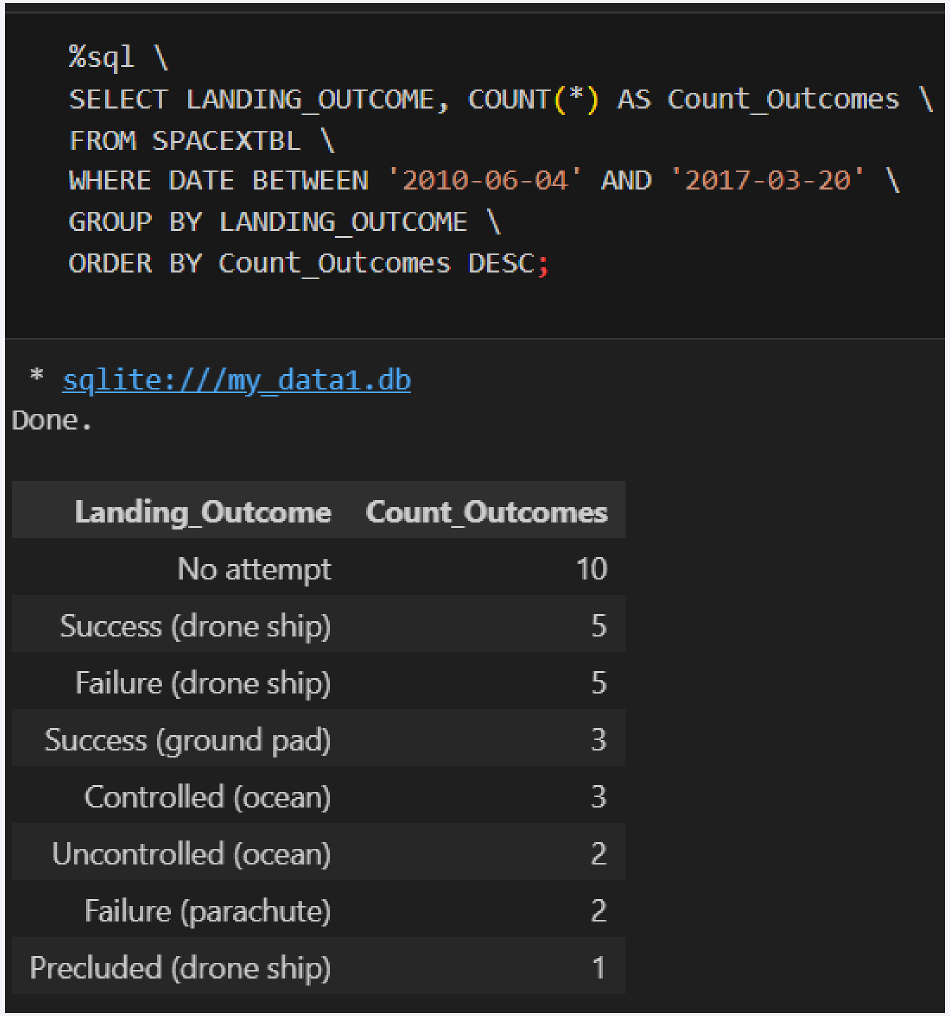


```
%sql \
SELECT SUBSTR(Date, 6, 2) AS Month, LANDING_OUTCOME, BOOSTER_VERSION, LAUNCH_SITE \
FROM SPACEXTBL \
WHERE LANDING_OUTCOME = 'Failure (drone ship)' AND SUBSTR(Date, 0, 5) = '2015';
```

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

Month	Landing_Outcome	Booster_Version	Launch_Site
01	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



```
%sql \
SELECT LANDING_OUTCOME, COUNT(*) AS Count_Outcomes \
FROM SPACEXTBL \
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' \
GROUP BY LANDING_OUTCOME \
ORDER BY Count_Outcomes DESC;
```

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

Landing_Outcome	Count_Outcomes
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

- 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

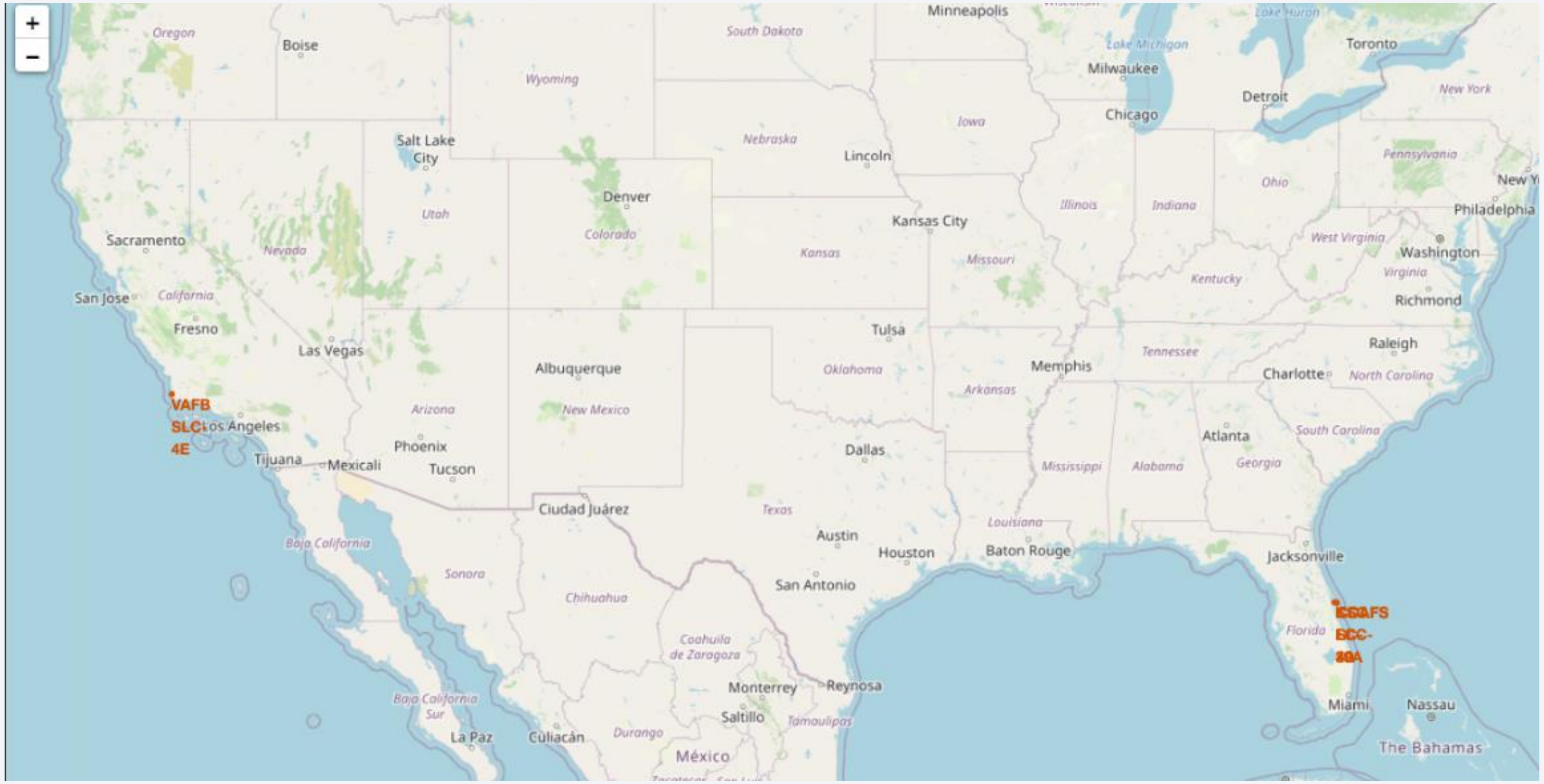
- Applying the GROUP BY clause, we grouped the landing outcomes, and utilized the ORDER BY clause to arrange the grouped landing outcome in descending order

Section 3

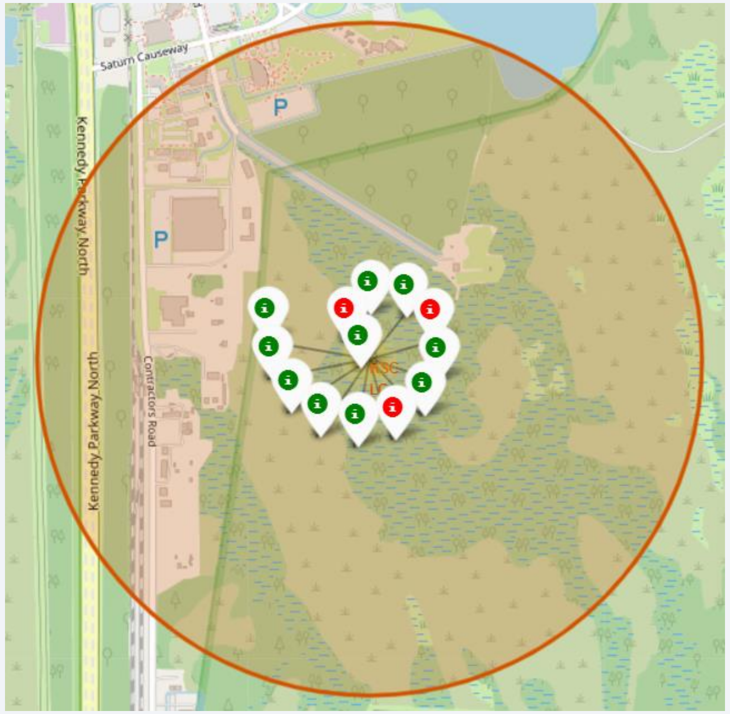
Launch Sites Proximities Analysis

Location of all Launch Sites

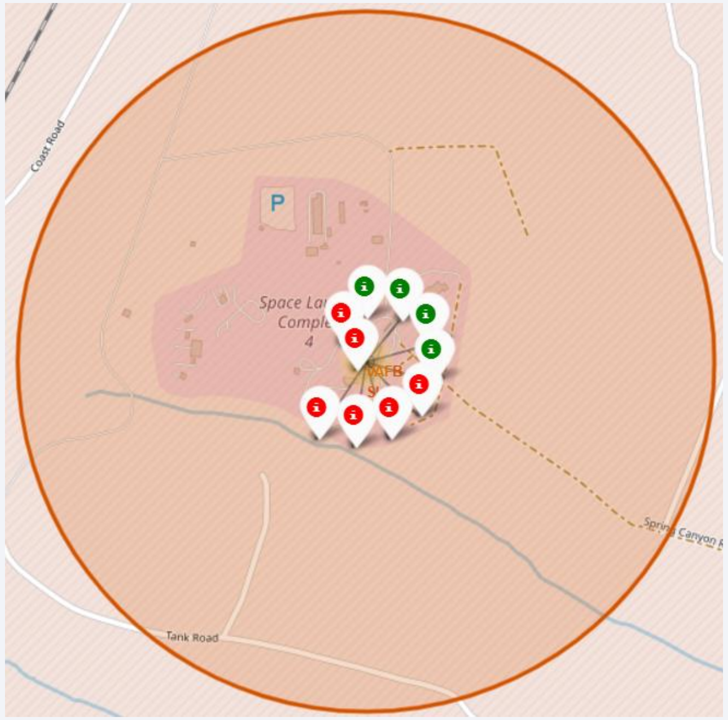
It's apparent that all SpaceX launch sites are situated within the United States



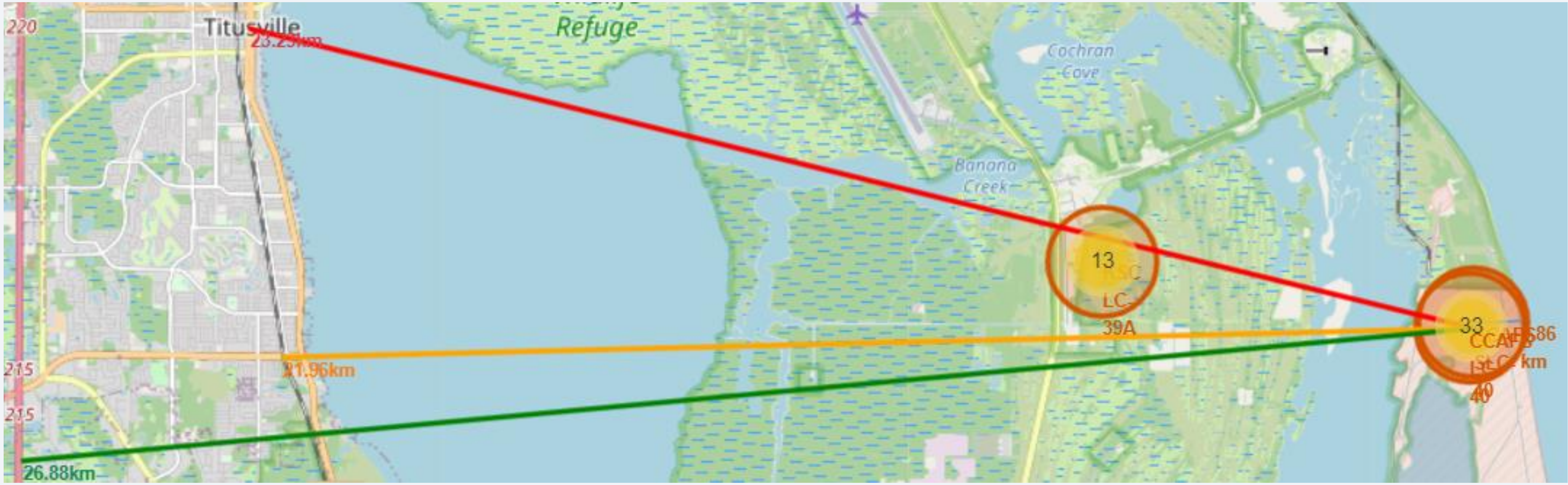
Markers Showing Launch Sites with Labels



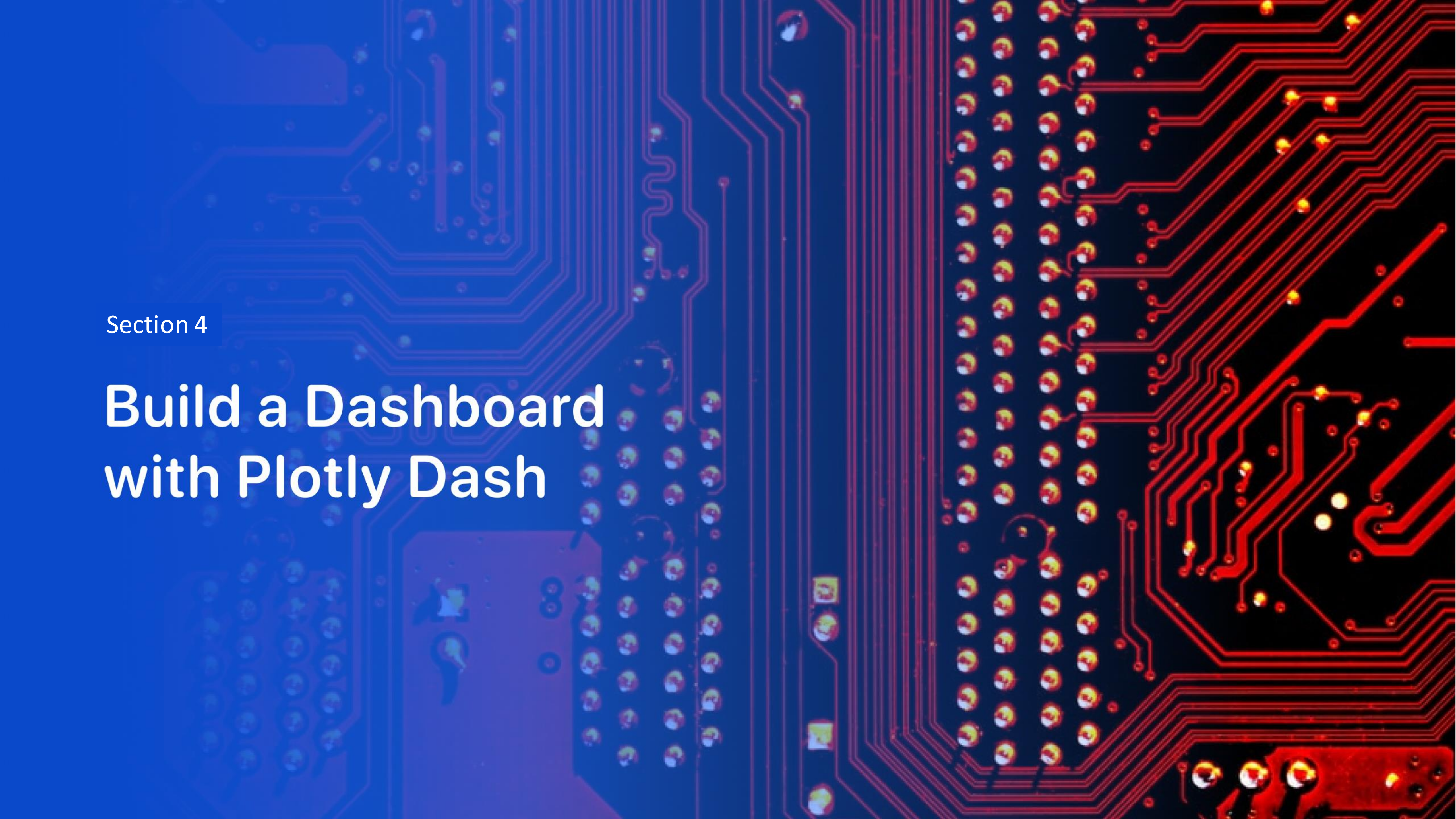
- Green markers – successful launches
- Red markers – unsuccessful launches



Launch Sites Distance to Selected Landmarks



Map showing various distances to selected landmarks

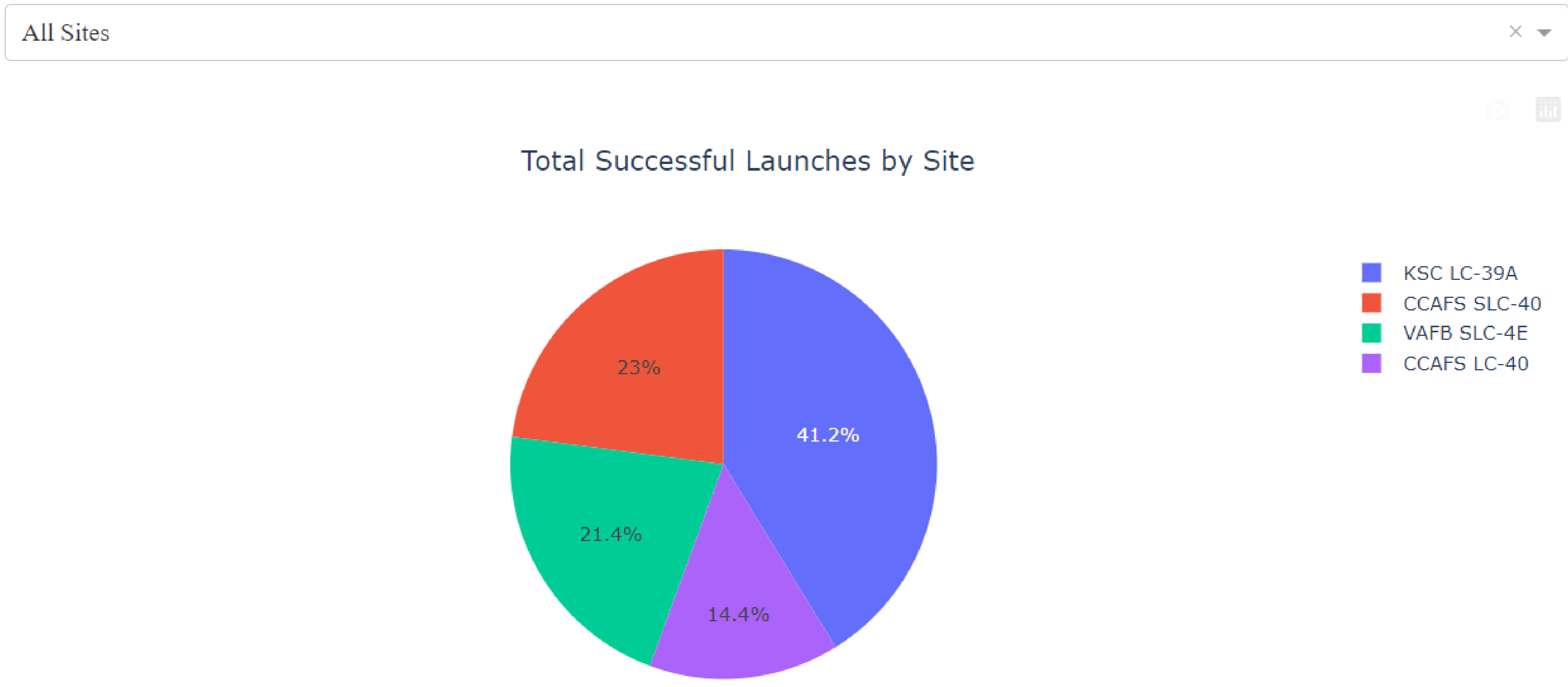


Section 4

Build a Dashboard with Plotly Dash

Pie Chart Illustrating the Success Percentage Attained by Each Launch Site

It's apparent that KSC LC-39A had the biggest percentage of successful launches

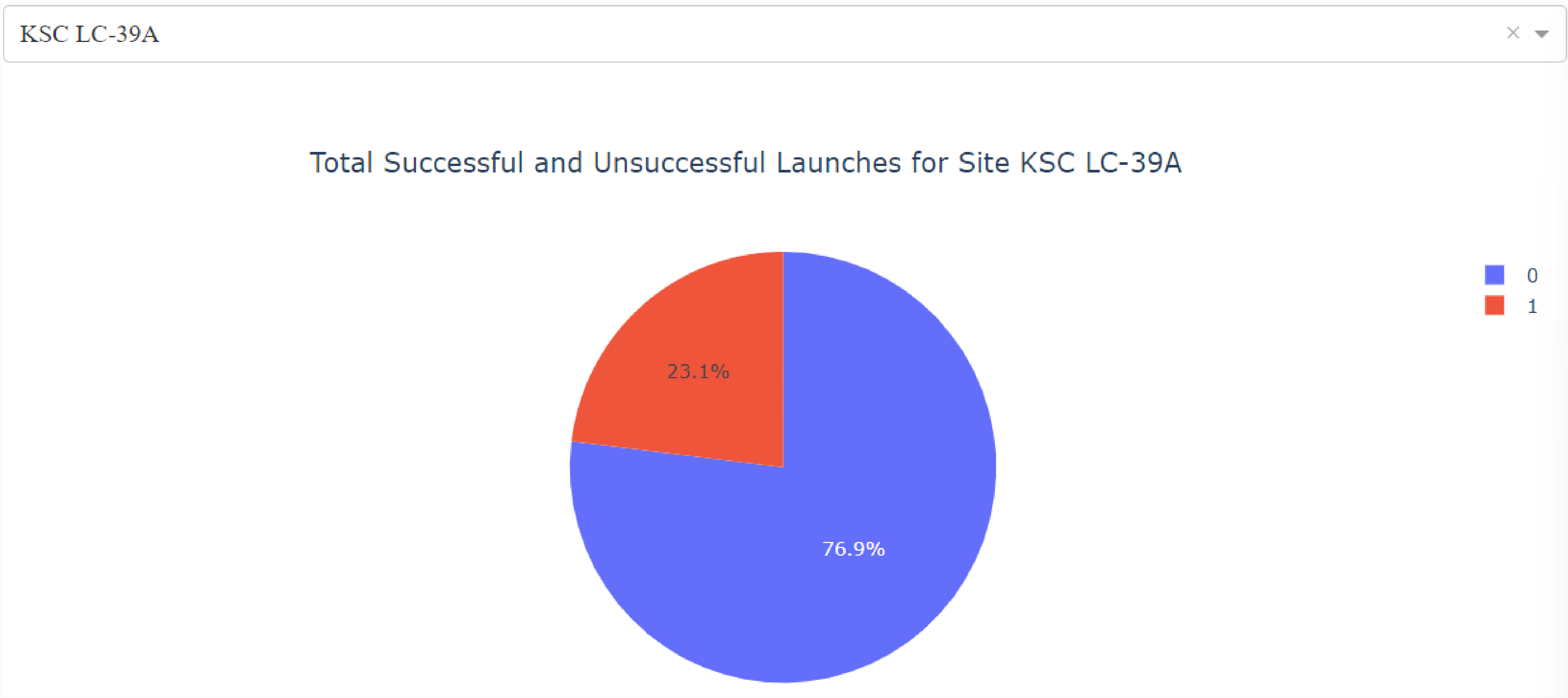


All Sites

Total Successful Launches by Site

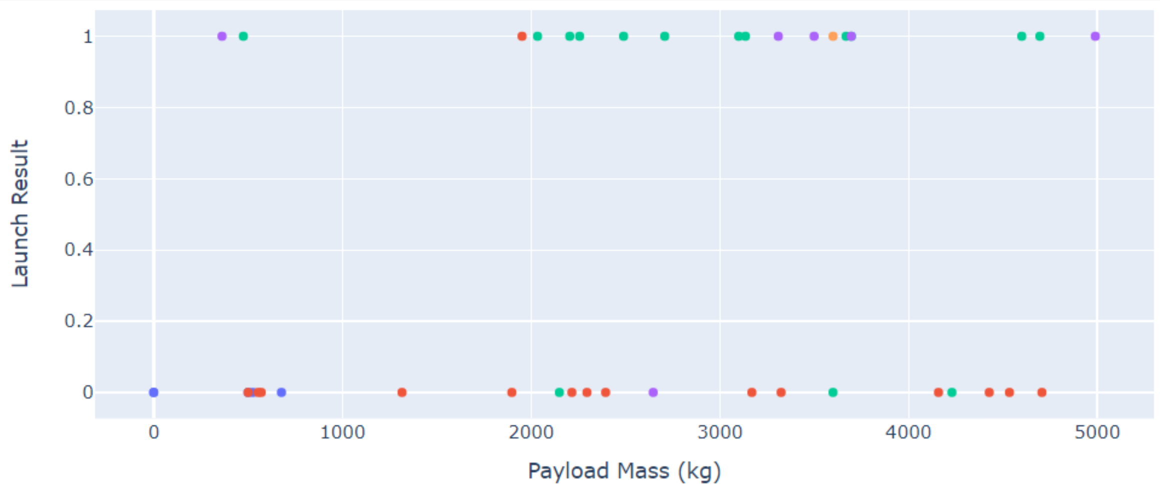
Pie Chart Illustrating the Success Percentage Attained by Each Launch Site

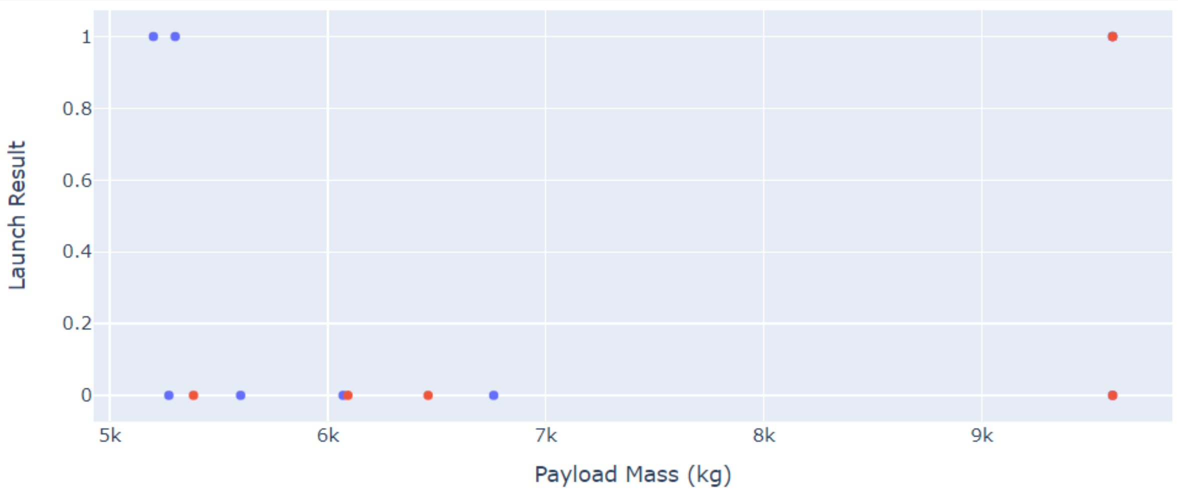
The site KSC LC-39A had nearly 77 % of successful launches


KSC LC-39A
×
Total Successful and Unsuccessful Launches for Site KSC LC-39A
23.1%
76.9%
🟦 0
🟥 1

Scatter Plot of Payload vs. Launch Outcome for All Sites (Selection of Payload Ranges)

Payload 0-5,000 kg vs. payload 5,000-10,000 kg





The success rate for lighter payloads appears to be higher than those of heavier payloads

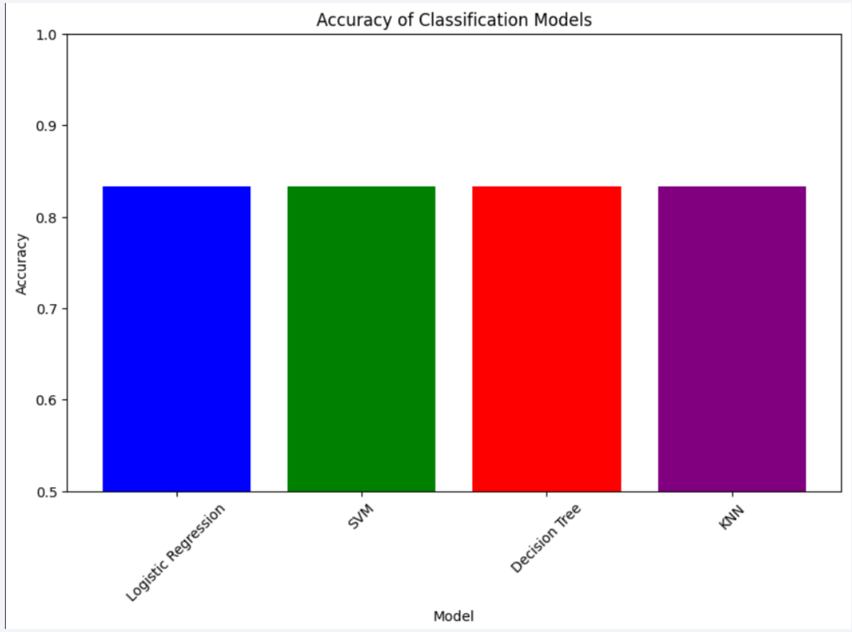
Section 5

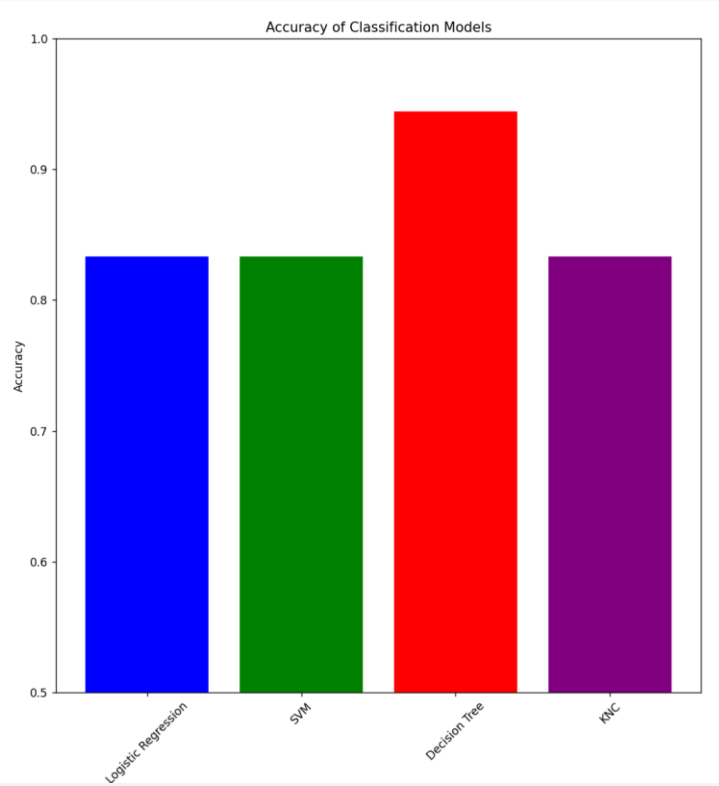
Predictive Analysis (Classification)

Classification Accuracy

In Jupyter Notebook all models have the same accuracy

→

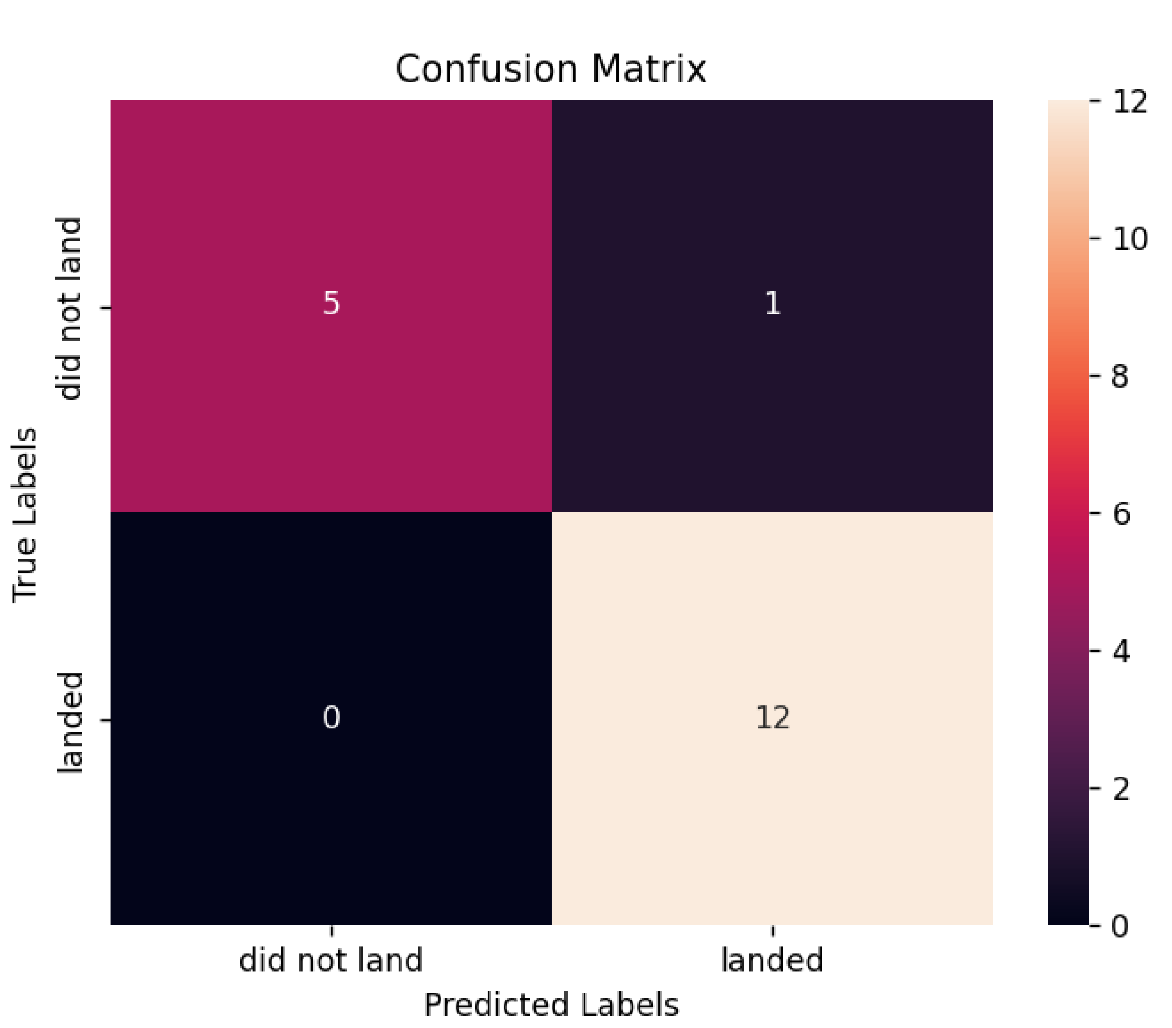




←

In pure python the decision tree model has the highest accuracy of nearly 95 %

Confusion Matrix



The confusion matrix generated for the decision tree classifier indicates its ability to differentiate between various classes. However, a notable issue arises with false positives, where unsuccessful landings are erroneously classified as successful landings by the classifier.

Conclusions

Based on the analysis, we can conclude the following:

- There is a positive correlation between the number of flights at a launch site and its success rate
- Launch success rates have exhibited a consistent upward trend from 2013 to 2020
- Orbits ES-L1, GEO, HEO, SSO, and VLEO have consistently demonstrated the highest success rates
- Among all launch sites, KSC LC-39A stands out with the highest number of successful launches
- The Decision Tree classifier emerges as the most effective machine learning algorithm for this task

Thank you!