IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Alpkan Öztürk
5th October 2023

Outline

Executive Summary

- Summary of methodologies
  - Data Collection with API
  - Data Collection with Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis with SQL
  - Exploratory Data Analysis with Data Visualization
  - Interactive Visual Analytics with Folium
  - Interactive Dashboard with Plotly Dash
  - Machine Learning Prediction
- Summary of all results
  - Exploratory Data Analysis Result
  - Interactive Analytics Results
  - Predictive Analysis Results

Introduction

- Project background and context

  SpaceX promotes Falcon 9 rocket launches for 62 million dollars; other suppliers charge upwards of 165 million dollars for each launch. A large portion of the savings is due to SpaceX's ability to reuse the first stage. So, if we can figure out whether the first stage will land, we can figure out how much a launch will cost. The Falcon 9 from Spaces X takes off like a typical rocket. If another business wishes to submit a proposal for a rocket launch against space X, they can use this information. The project's objective is to build a pipeline for machine learning that can forecast if the initial stage will land successfully.

- Problems you want to find answers
  - What elements determine whether the rocket will successfully land?
  - The interaction between different features that affects the likelihood of a successful landing.
  - What operational requirements must be met to guarantee a successful landing program?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using SpaceX API and web scraping through the Wikipedia.
- Perform data wrangling
  - Categorical information was converted to binary system using the one-hot encoding method.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Machine learning algorithms were used to perform Predictive Analysis.

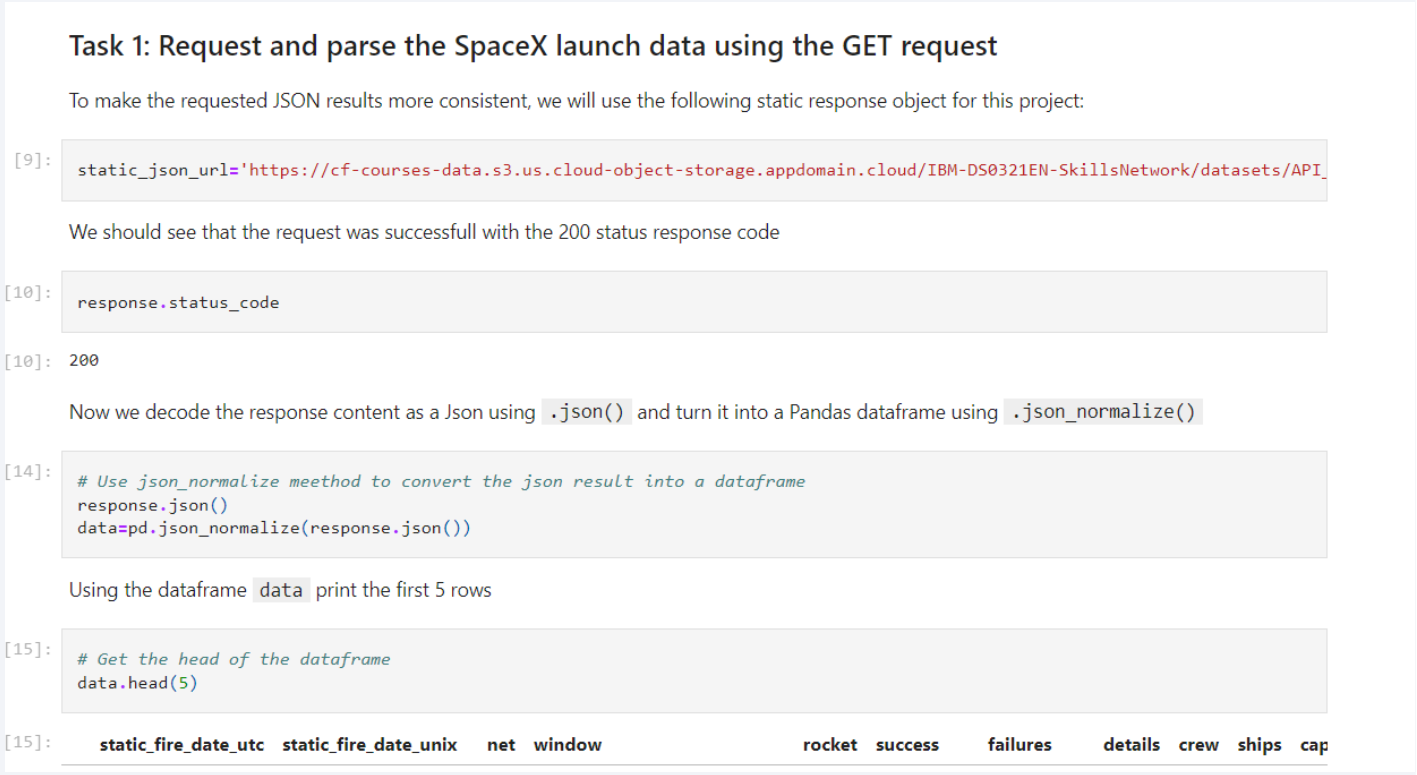
Data Collection

- Data collected using SpaceX API.
  - We used the.json() function call to decode the response's content as JSON and the .json_normalize() method to convert it to a pandas dataframe.
  - Next, we proceeded to tidy up the data, scrutinizing it for any absent information and addressing those gaps by providing necessary values when required.
- Data collected using web scraping through the Wikipedia.
  - BeautifulSoup library was used.
  - Goal was to retrieve the launch records in the form of an HTML table, parse that table, and transform it into a pandas dataframe for subsequent analysis.

Data Collection – SpaceX API

- We employed a get request to obtain data from the SpaceX API, then proceeded to refine the acquired data by performing essential data cleaning and engaging in some fundamental data manipulation and formatting.

- Notebook Link: https://github.com/alpkanoz/Exploratary_Data_Analysis_Falcon_9_Landing/blob/d8d5620f91b31e2acad2d1ba5f3901f4bc0d5010/jupyter-labs-spacex-data-collection-api.ipynb



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

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

```
[9]: static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_
```

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()`

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

Using the dataframe `data` print the first 5 rows

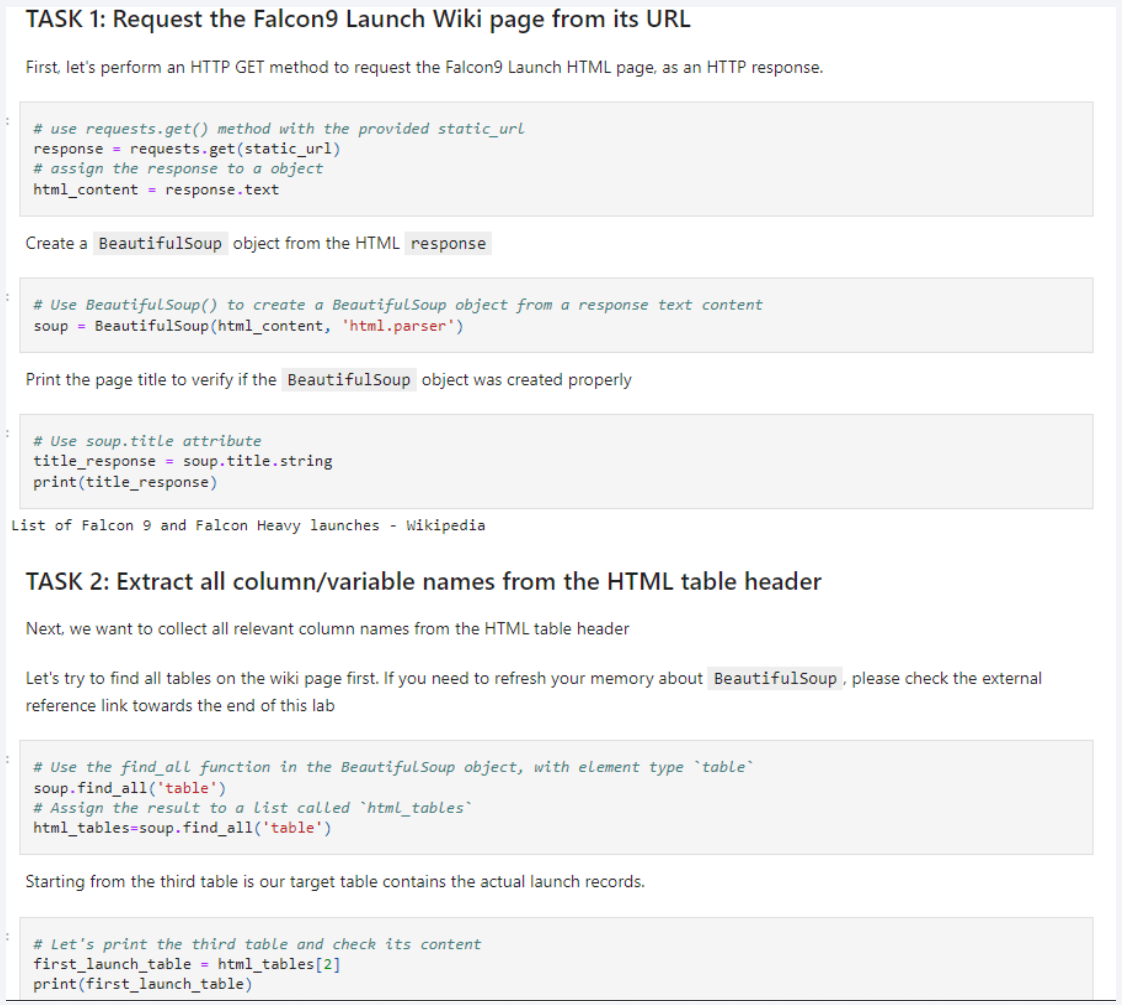
```
[15]: # Get the head of the dataframe
      data.head(5)
```

```
[15]:   static_fire_date_utc  static_fire_date_unix  net  window            rocket  success    failures    details  crew  ships  cap
```

Data Collection - Scraping

- We utilized web scraping techniques using BeautifulSoup to extract Falcon 9 launch records from Wikipedia. Subsequently, we parsed the table data and transformed it into a pandas dataframe for further analysis.

- https://github.com/alpkanoz/Exploratary_Data_Analysis_Falcon_9_Landing/blob/d8d5620f91b31e2acad2d1ba5f3901f4bc0d5010/jupyter-labs-webscraping.ipynb



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.

```python
# use requests.get() method with the provided static_url
response = requests.get(static_url)
# assign the response to a object
html_content = response.text
```

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

```python
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(html_content, 'html.parser')
```

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

```python
# Use soup.title attribute
title_response = soup.title.string
print(title_response)
```

```
List of Falcon 9 and Falcon Heavy launches - Wikipedia
```

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

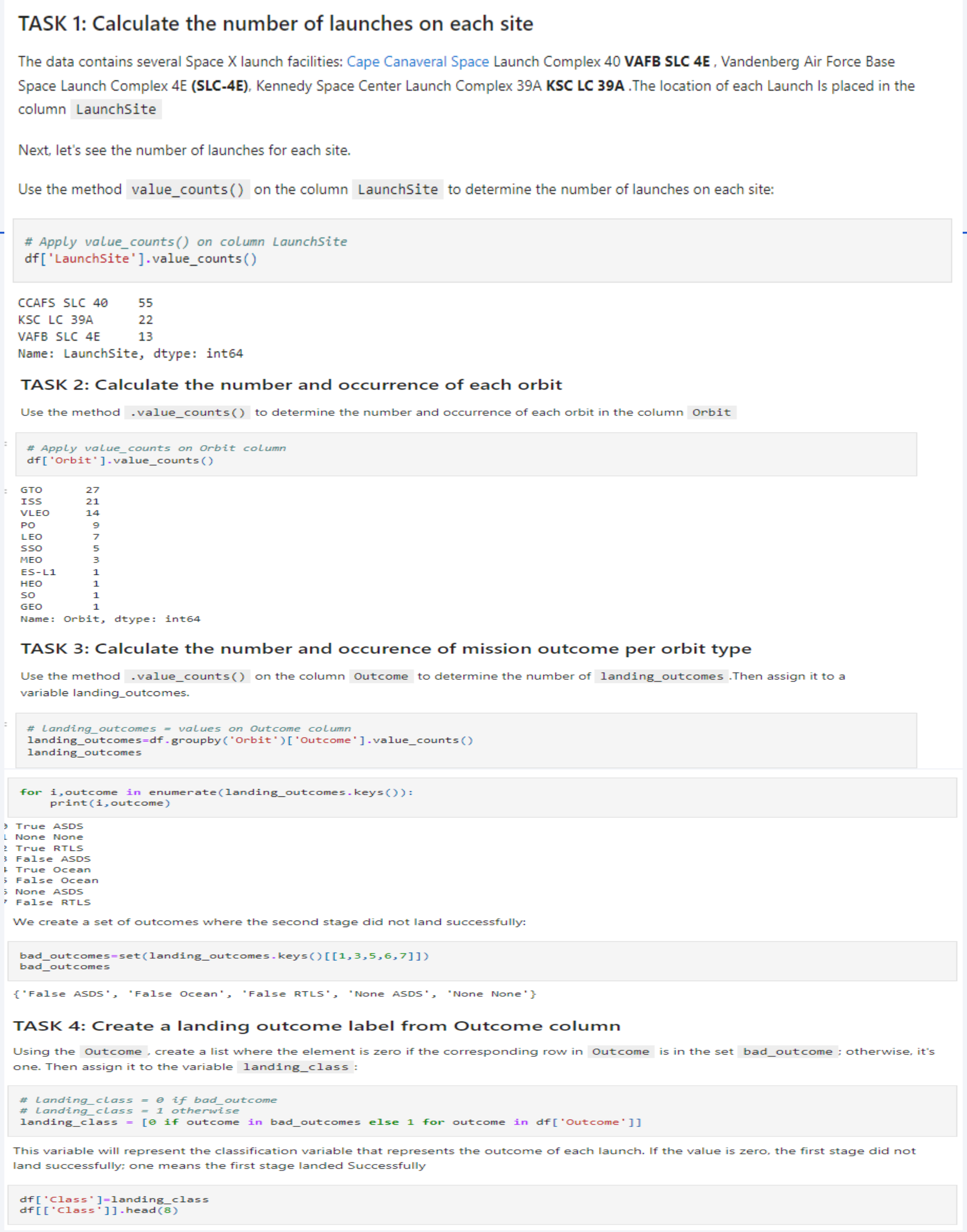
```python
# Use the find_all function in the BeautifulSoup object, with element type `table`
soup.find_all('table')
# Assign the result to a list called `html_tables`
html_tables=soup.find_all('table')
```

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

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

Data Wrangling

- We conducted exploratory data analysis to establish the training labels. This involved computing the count of launches at each site, as well as quantifying and documenting the frequency of different orbits. Furthermore, we generated a landing outcome label based on the outcome column and then saved the results to a CSV file.
- Notebook Link : https://github.com/alpkanoz/Exploratary_Data_Analysis_Falcon_9_Landing/blob/d8d5620f91b31e2acad2d1ba5f3901f4bc0d5010/labs-jupyter-spacex-data_wrangling_jupyterlite.ipynb



TASK 1: Calculate the number of launches on each site

The data contains several Space X launch facilities: Cape Canaveral Space Launch Complex 40 **VAFB SLC 4E** , Vandenberg Air Force Base Space Launch Complex 4E **(SLC-4E)**, Kennedy Space Center Launch Complex 39A **KSC LC 39A** .The location of each Launch Is placed in the column `LaunchSite`

Next, let's see the number of launches for each site.

Use the method `value_counts()` on the column `LaunchSite` to determine the number of launches on each site:

```
# Apply value_counts() on column LaunchSite
df['LaunchSite'].value_counts()
```

```
CCAFS SLC 40    55
KSC LC 39A      22
VAFB SLC 4E     13
Name: LaunchSite, dtype: int64
```

TASK 2: Calculate the number and occurrence of each orbit

Use the method `.value_counts()` to determine the number and occurrence of each orbit in the column `Orbit`

```
# Apply value_counts on Orbit column
df['Orbit'].value_counts()
```

```
GTO      27
ISS      21
VLEO     14
PO        9
LEO       7
SSO       5
MEO       3
ES-L1     1
HEO       1
SO        1
GEO       1
Name: Orbit, dtype: int64
```

TASK 3: Calculate the number and occurence of mission outcome per orbit type

Use the method `.value_counts()` on the column `Outcome` to determine the number of `landing_outcomes` .Then assign it to a variable landing_outcomes.

```
# landing_outcomes = values on Outcome column
landing_outcomes=df.groupby('Orbit')['Outcome'].value_counts()
landing_outcomes
```

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

```
True ASDS
None None
True RTLS
False ASDS
True Ocean
False Ocean
None ASDS
False RTLS
```

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

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

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

TASK 4: Create a landing outcome label from Outcome column

Using the `Outcome` , create a list where the element is zero if the corresponding row in `Outcome` is in the set `bad_outcome` ; otherwise, it's one. Then assign it to the variable `landing_class` :

```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
landing_class = [0 if outcome in bad_outcomes else 1 for outcome in df['Outcome']]
```

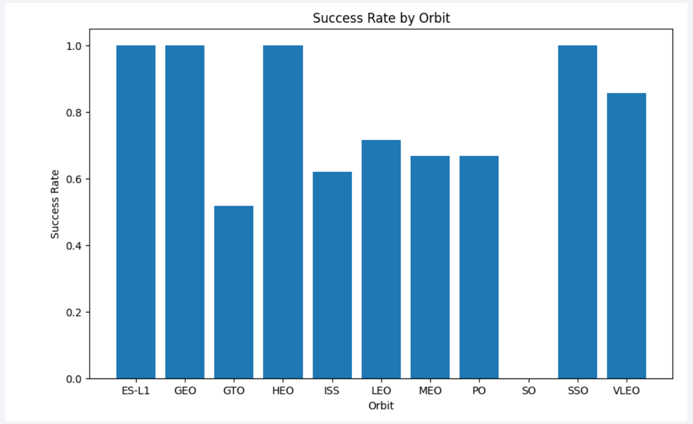
This variable will represent the classification variable that represents the outcome of each launch. If the value is zero, the first stage did not land successfully; one means the first stage landed Successfully

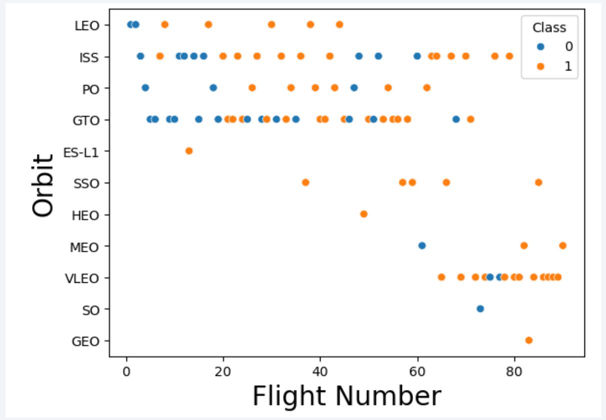
```
df['Class']=landing_class
df[['Class']].head(8)
```

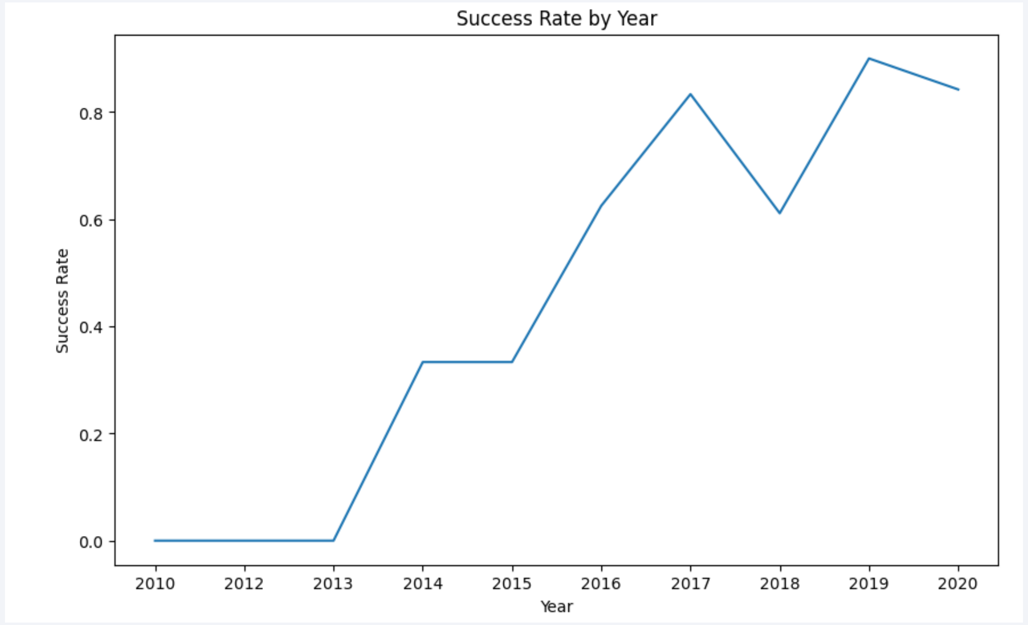
EDA with Data Visualization

- We delved into the data by creating visual representations to examine the connections between flight number and launch site, payload and launch site, the success rate of various orbit types, flight number and orbit type, as well as the annual trend in launch success.

- Notebook Link: https://github.com/alpkanoz/Exploratary_Data_Analysis_Falcon_9_Landing/blob/d8d5620f91b31e2acad2d1ba5f3901f4bc0d5010/jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb







EDA with SQL

- We seamlessly imported the SpaceX dataset into a PostgreSQL database directly within the Jupyter notebook environment. We then utilized SQL for Exploratory Data Analysis (EDA) to gain valuable insights from the dataset. Through SQL queries, we sought information on various aspects such as;
    - The average payload mass carried by booster version F9 v1.1.
    - The total number of successful and failure mission outcomes.
    - Counting the total number of successful and failed mission outcomes.
    - Identifying the distinct names of launch sites.
- Notebook Link: https://github.com/alpkanoz/Exploratary_Data_Analysis_Falcon_9_Landing/blob/818f95bfdacd7163a6005ba3c026013f81193f8a/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- We annotated all launch sites on the map and incorporated map elements such as markers, circles, and lines to visually represent the success or failure of launches for each site using Folium.
- We assigned a binary feature to launch outcomes, where '0' indicates failure and '1' indicates success.
- By utilizing color-coded marker clusters, we pinpointed launch sites with notably high success rates.
- We computed the distances between launch sites and their surrounding areas and addressed inquiries such as:
  - Are launch sites located in close proximity to railways, highways, or coastlines?
  - Do launch sites maintain a certain distance from urban areas?
- Notebook Link: https://github.com/alpkanoz/Interactive_Visual_Analytics_and_Dashboard_Falcon9_Landing/blob/a74421babb9887ff643a6fbeb0537517719dd1b3/lab_jupyter_launch_site_location.jupyterlite.ipynb

Build a Dashboard with Plotly Dash

- We constructed an interactive dashboard using Plotly Dash.
- We generated pie charts to visualize the overall number of launches from specific sites.
- We created scatter plots to depict the correlation between Outcome and Payload Mass (Kg) for various booster versions.
- Dash App Link: https://github.com/alpkanoz/Interactive_Visual_Analytics_and_Dashboard_Falcon9_Landing/blob/a74421babb9887ff643a6fbeb0537517719dd1b3/spacex_dash_app.py

Predictive Analysis (Classification)

- We imported the data utilizing NumPy and Pandas, conducted data transformation, and divided it into training and testing sets.
- We constructed multiple machine learning models and fine-tuned various hyperparameters using GridSearchCV.
- We employed accuracy as the primary metric for model evaluation and enhanced the model through feature engineering and algorithm tuning.
- We identified the top-performing classification model.
- Notebook Link: https://github.com/alpkanoz/Predictive_Analysis_classification/blob/001a0b38f0f9dd58d404efd90e92250f0a4ad436/SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

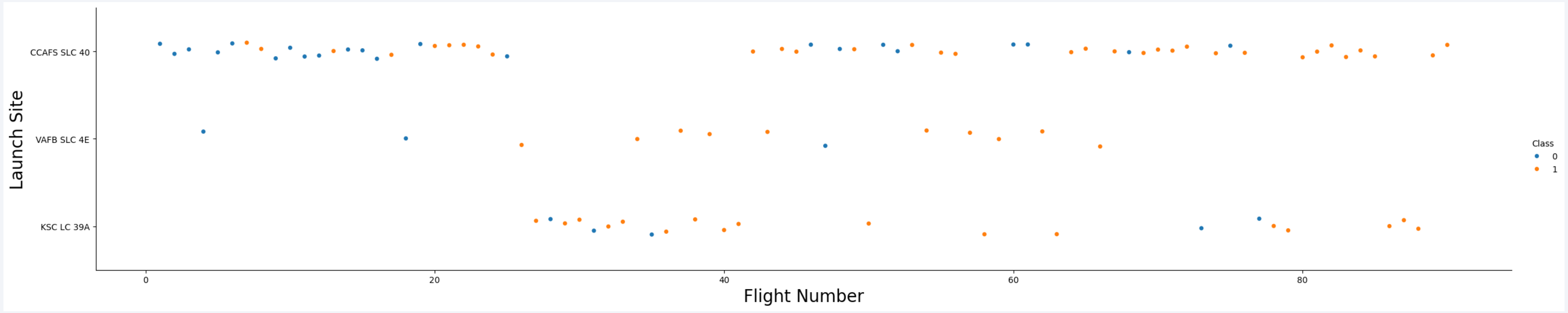
Results

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

Section 2

Insights drawn from EDA

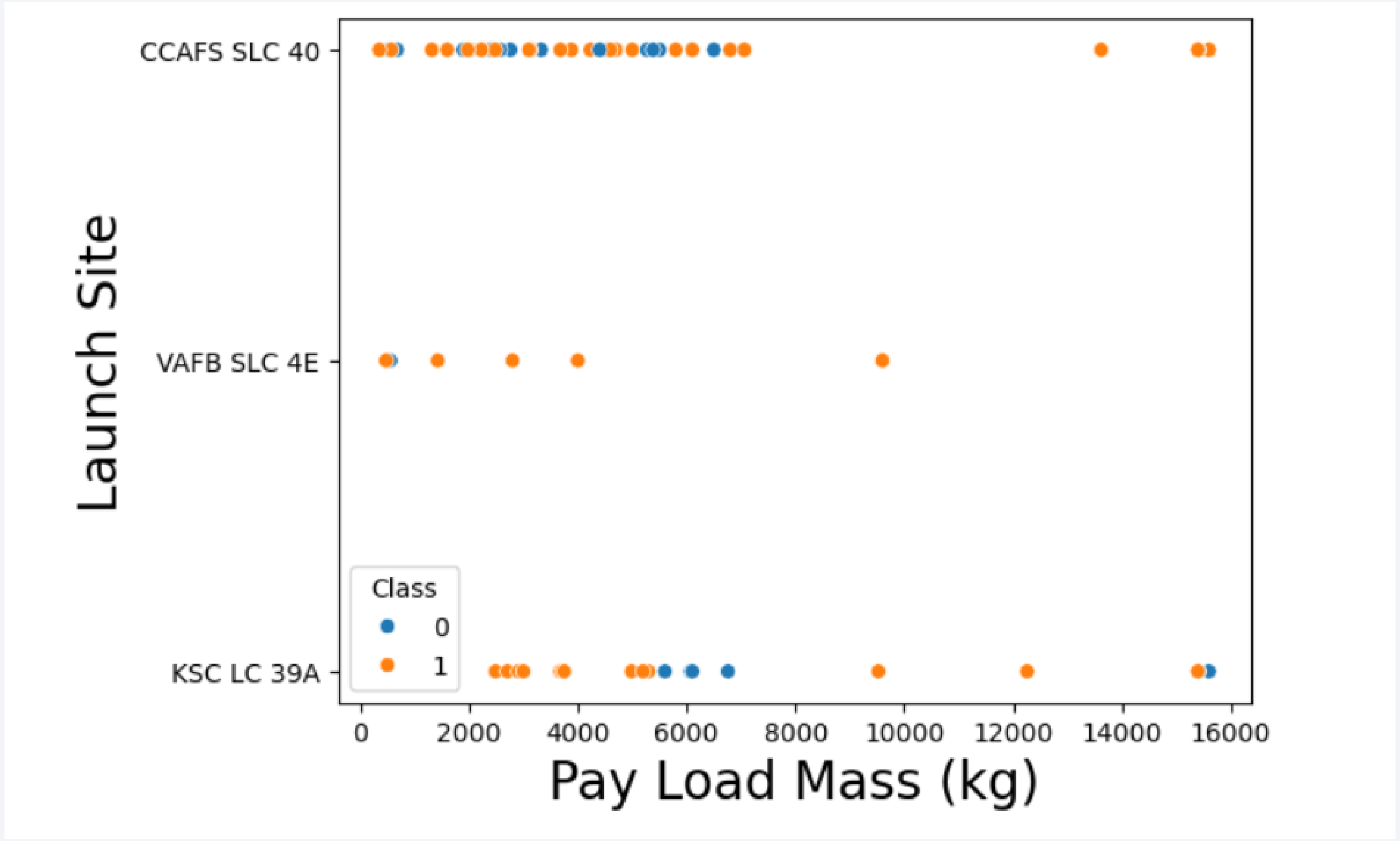
Flight Number vs. Launch Site



- As the number of flights per launch area increased, the success rate also increased.

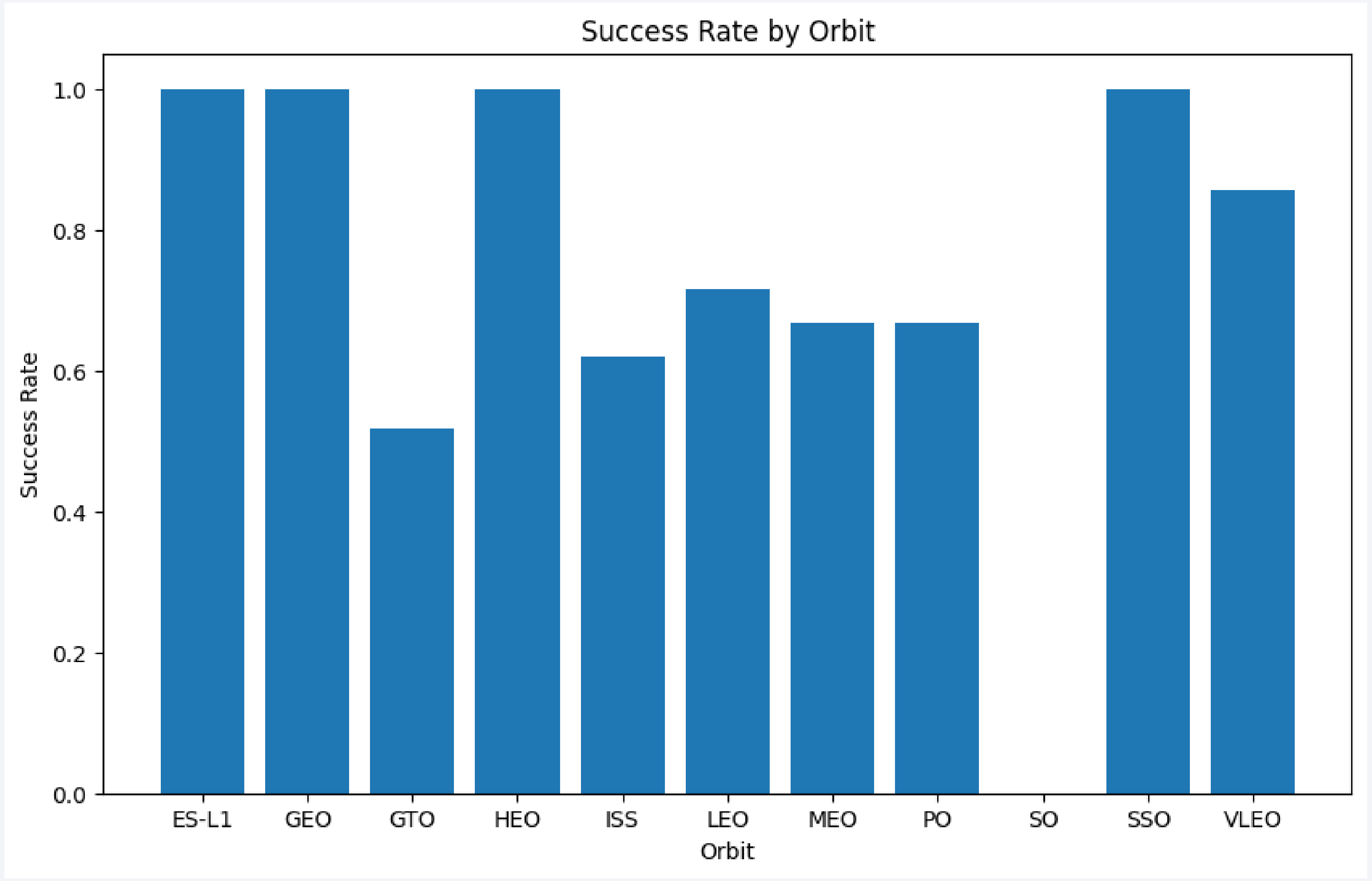
Payload vs. Launch Site

- At the VAFB-SLC launch site, there have been no rocket launches with a heavy payload mass exceeding 10,000.



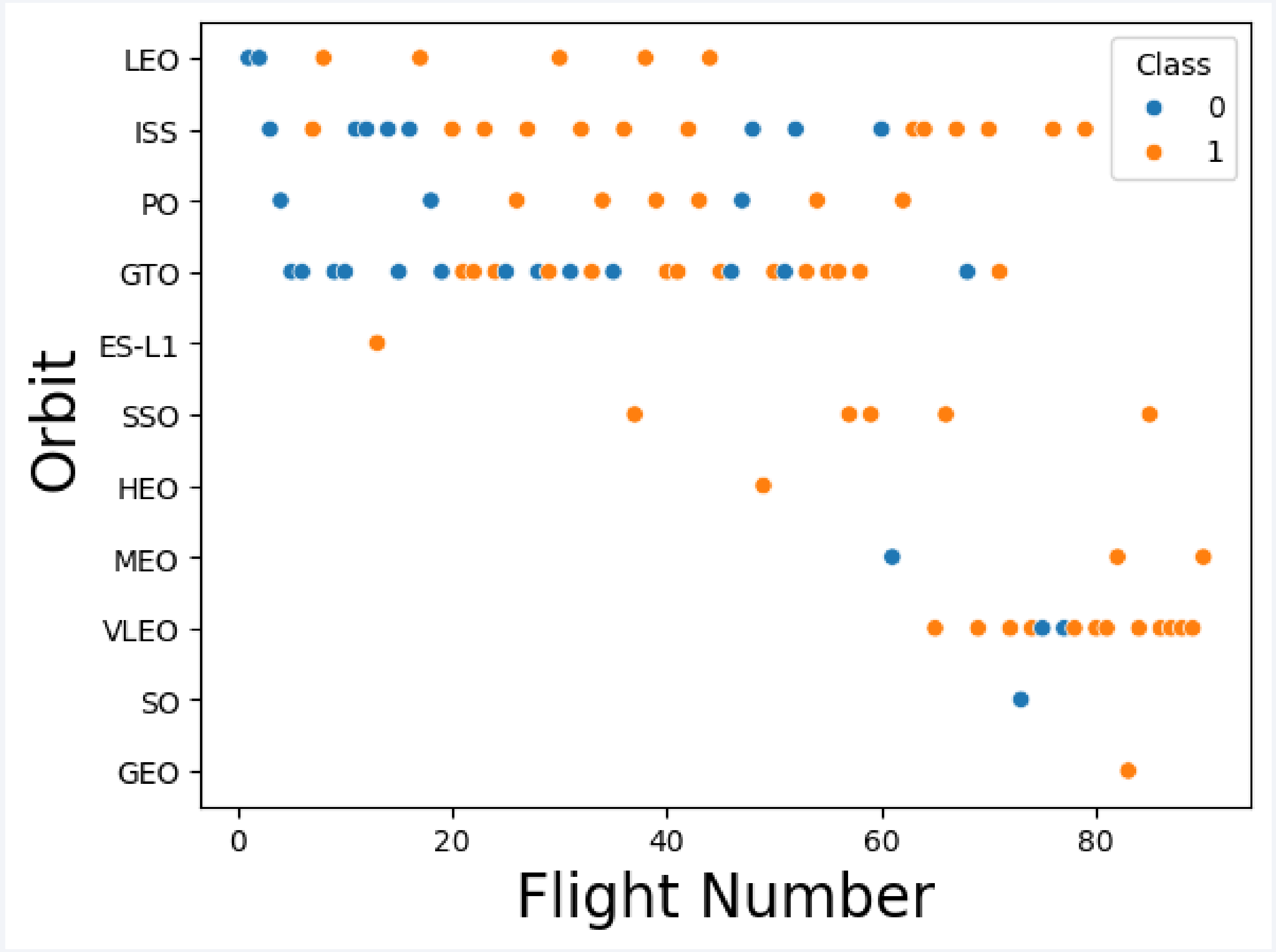
Success Rate vs. Orbit Type

- ES-L1, GEO, HEO, SSO orbits are the ones with the highest success rates.



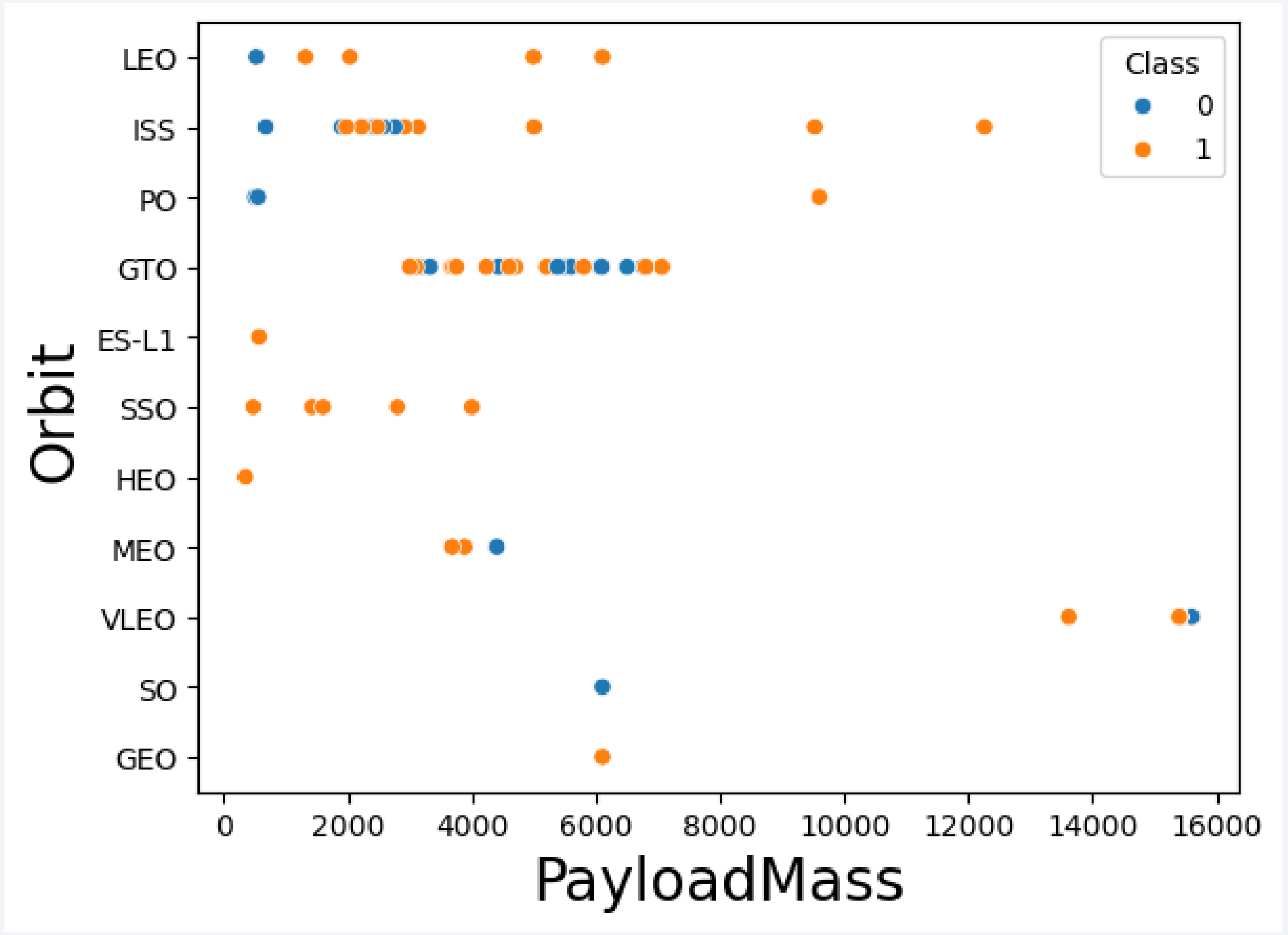
Flight Number vs. Orbit Type

- In Low Earth Orbit (LEO), the success rate seems to be connected to the frequency of flights, whereas in Geostationary Transfer Orbit (GTO), there doesn't appear to be any correlation between the flight number and success.



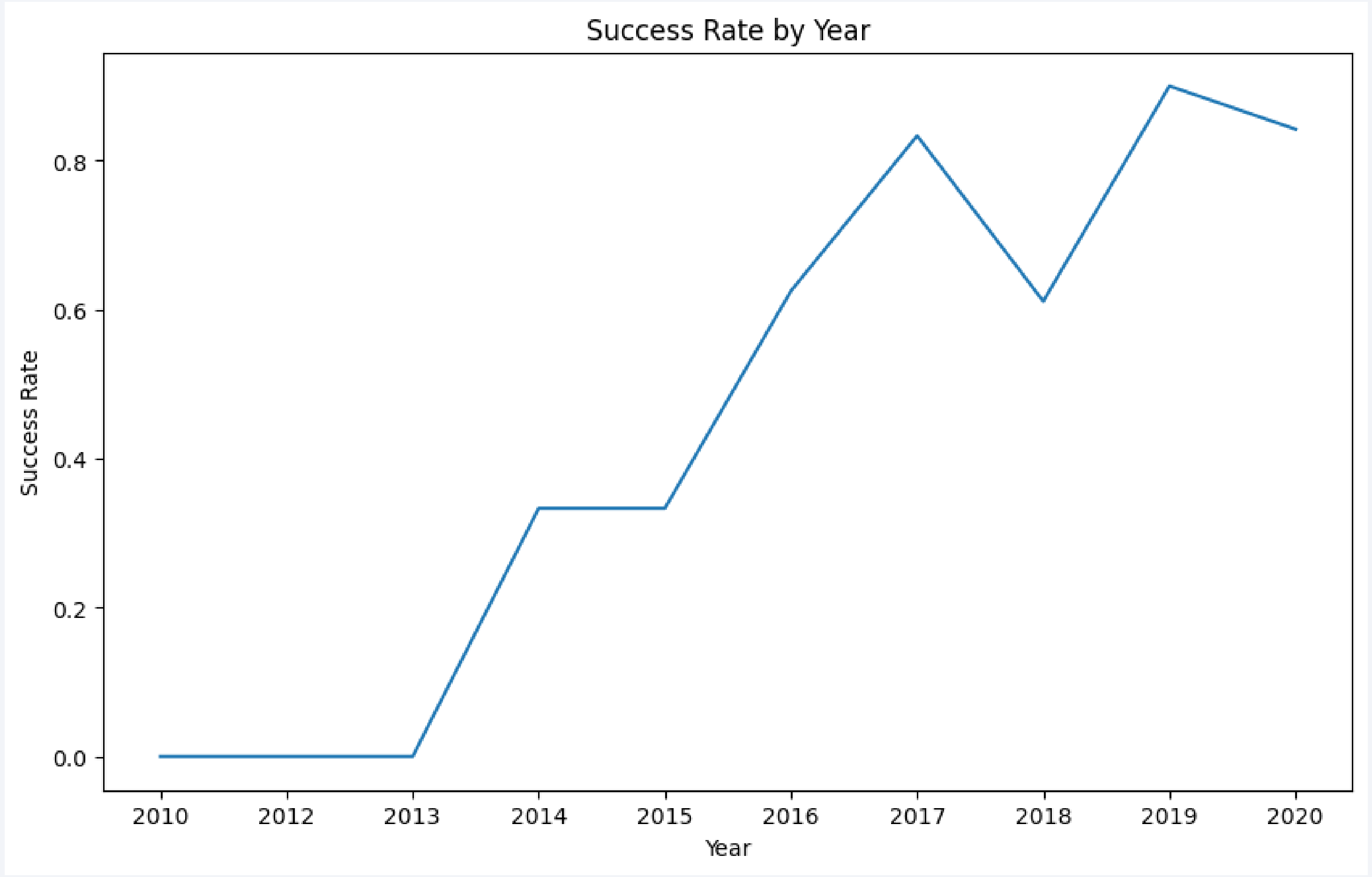
Payload vs. Orbit Type

- When dealing with heavy payloads, the likelihood of successful landings or positive landing rates is notably higher for missions in Polar, Low Earth Orbit (LEO), and the International Space Station (ISS) categories. However, in the case of Geostationary Transfer Orbit (GTO), it's challenging to clearly differentiate, as both positive landing rates and negative landing outcomes (unsuccessful missions) coexist.



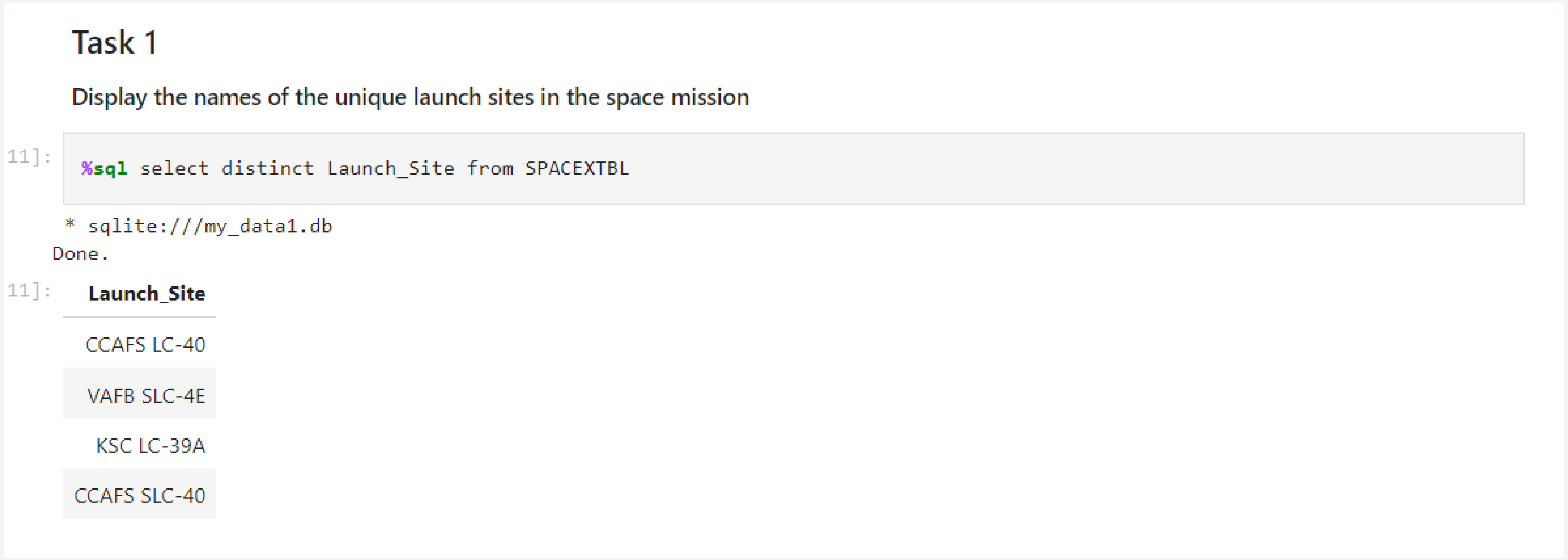
Launch Success Yearly Trend

- It's noticeable that the success rate has been on a continuous rise from 2013 up until the year 2020.



All Launch Site Names

- We used the key word DISTINCT to show only unique launch site names.



Task 1

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

```
11]: %sql select distinct Launch_Site from SPACEXTBL
```

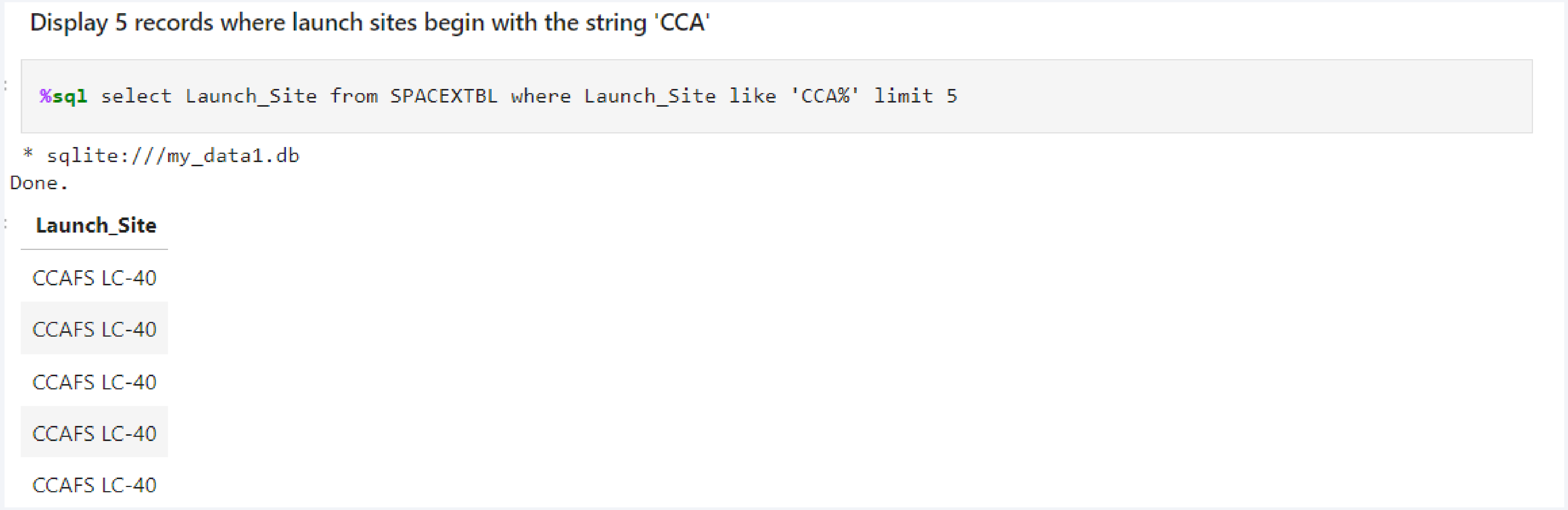
```
 * sqlite:///my_data1.db
Done.
```

11]:

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

Launch Site Names Begin with 'CCA'

- We used to query below the list launch sites which names begin with 'CCA'.



Display 5 records where launch sites begin with the string 'CCA'

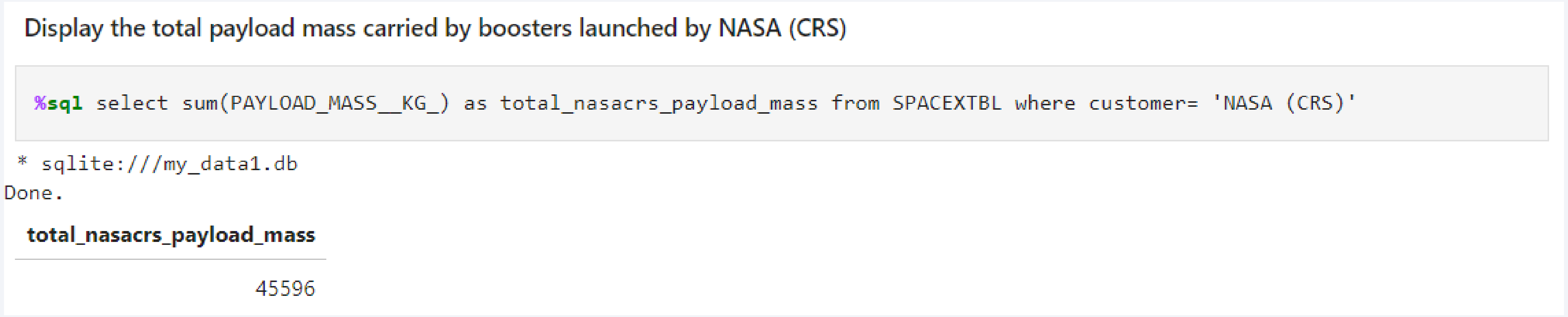
```
%sql select Launch_Site from SPACEXTBL where Launch_Site like 'CCA%' limit 5
```

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

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

Total Payload Mass

- We used the query in below to calculate the total payload carried by boosters from NASA.



Display the total payload mass carried by boosters launched by NASA (CRS)

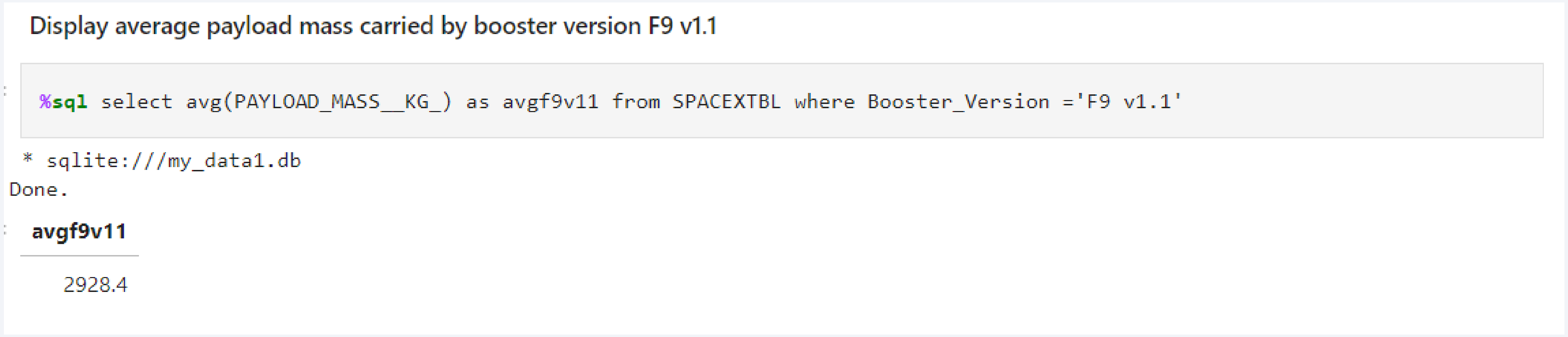
```
%sql select sum(PAYLOAD_MASS__KG_) as total_nasacrs_payload_mass from SPACEXTBL where customer= 'NASA (CRS)'
```

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

total_nasacrs_payload_mass
45596

Average Payload Mass by F9 v1.1

- We used the query in below to calculate the average payload mass carried by booster version F9 v1.1.



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

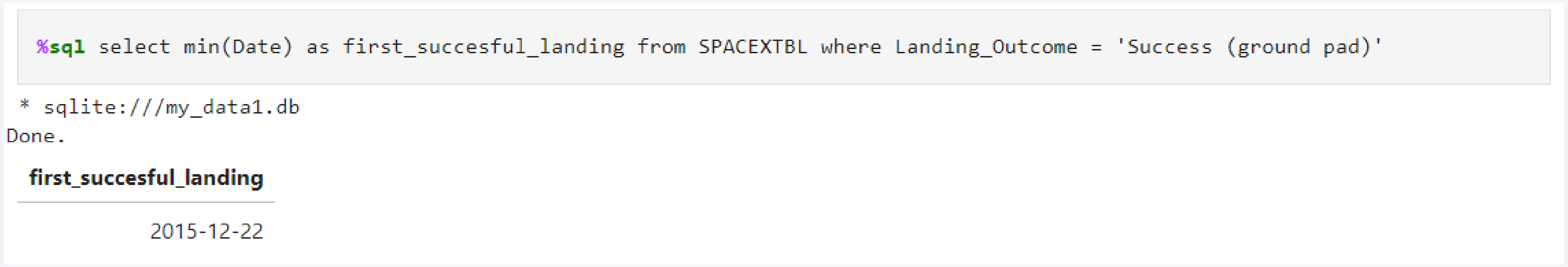
```
%sql select avg(PAYLOAD_MASS__KG_) as avgf9v11 from SPACEXTBL where Booster_Version ='F9 v1.1'
```

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

avgf9v11
2928.4

First Successful Ground Landing Date

- We used the query below to find the dates of the first successful landing outcome on ground pad.



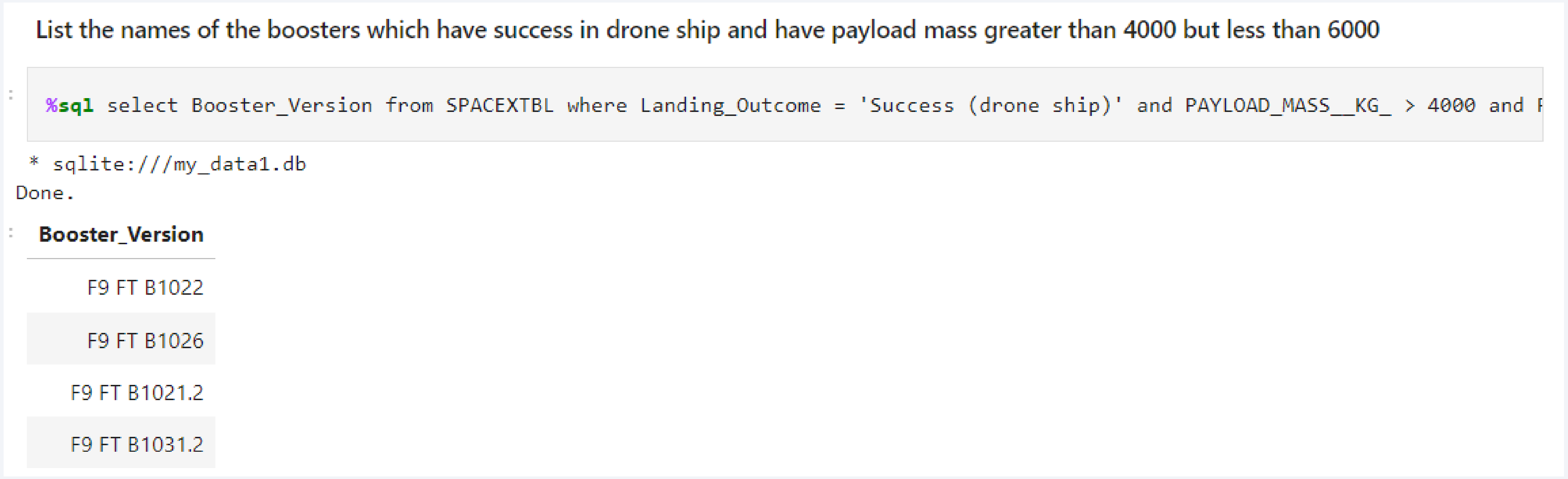
```
%sql select min(Date) as first_succesful_landing from SPACEXTBL where Landing_Outcome = 'Success (ground pad)'
```

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

first_succesful_landing
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- We used the query below to list the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000.



List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

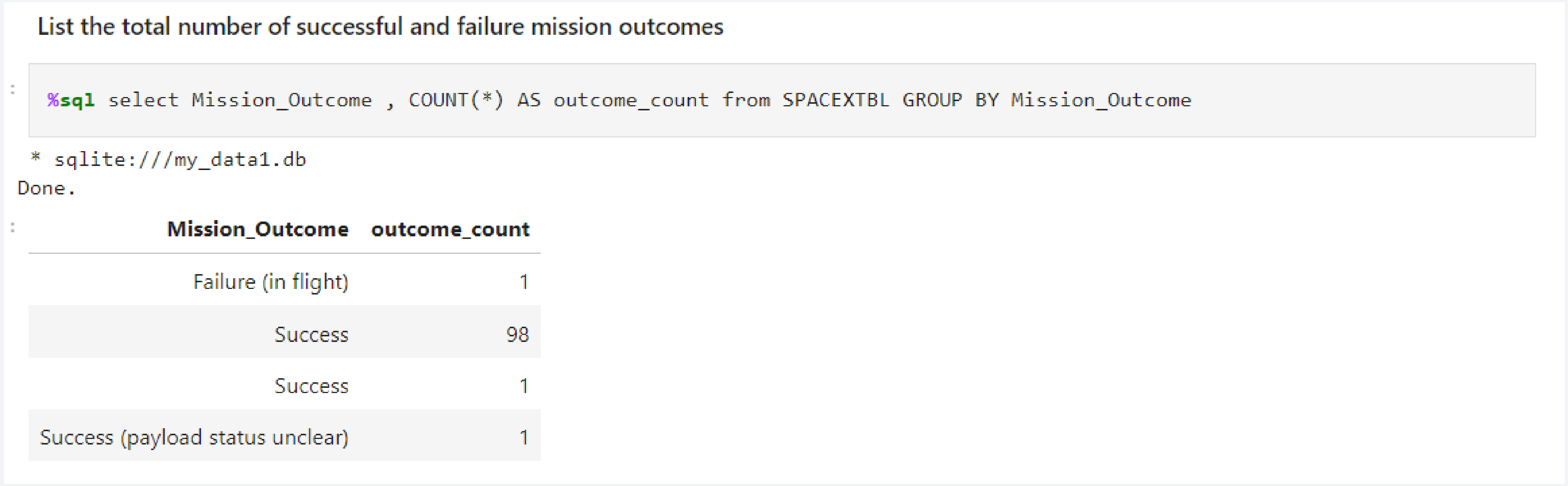
```
%sql select Booster_Version from SPACEXTBL where Landing_Outcome = 'Success (drone ship)' and PAYLOAD_MASS__KG_ > 4000 and P
```

```
 * 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

- We used the query in below to calculate the total number of successful and failure mission outcomes.



List the total number of successful and failure mission outcomes

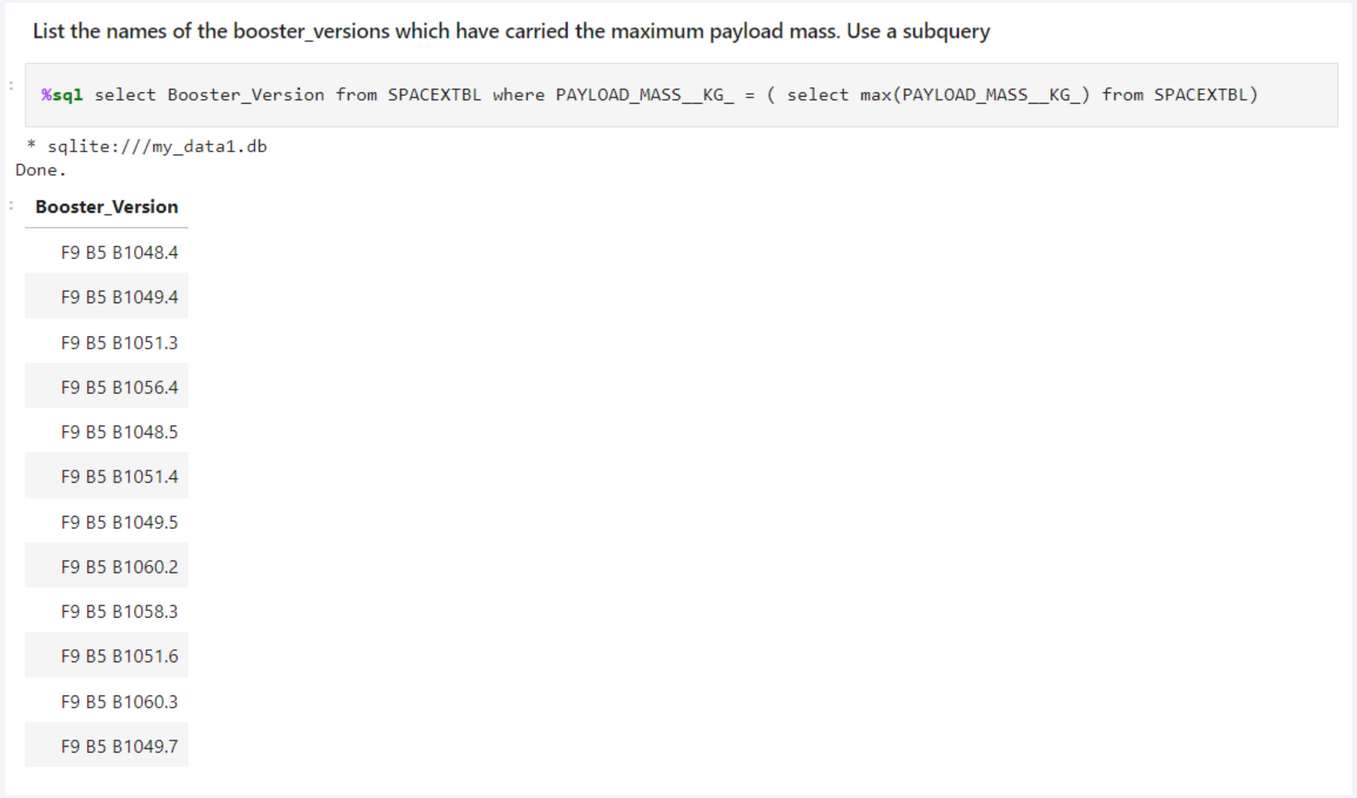
```
%sql select Mission_Outcome , COUNT(*) AS outcome_count from SPACEXTBL GROUP BY Mission_Outcome
```

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

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

Boosters Carried Maximum Payload

- We used the query in below to list the names of the booster which have carried the maximum payload mass.



List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

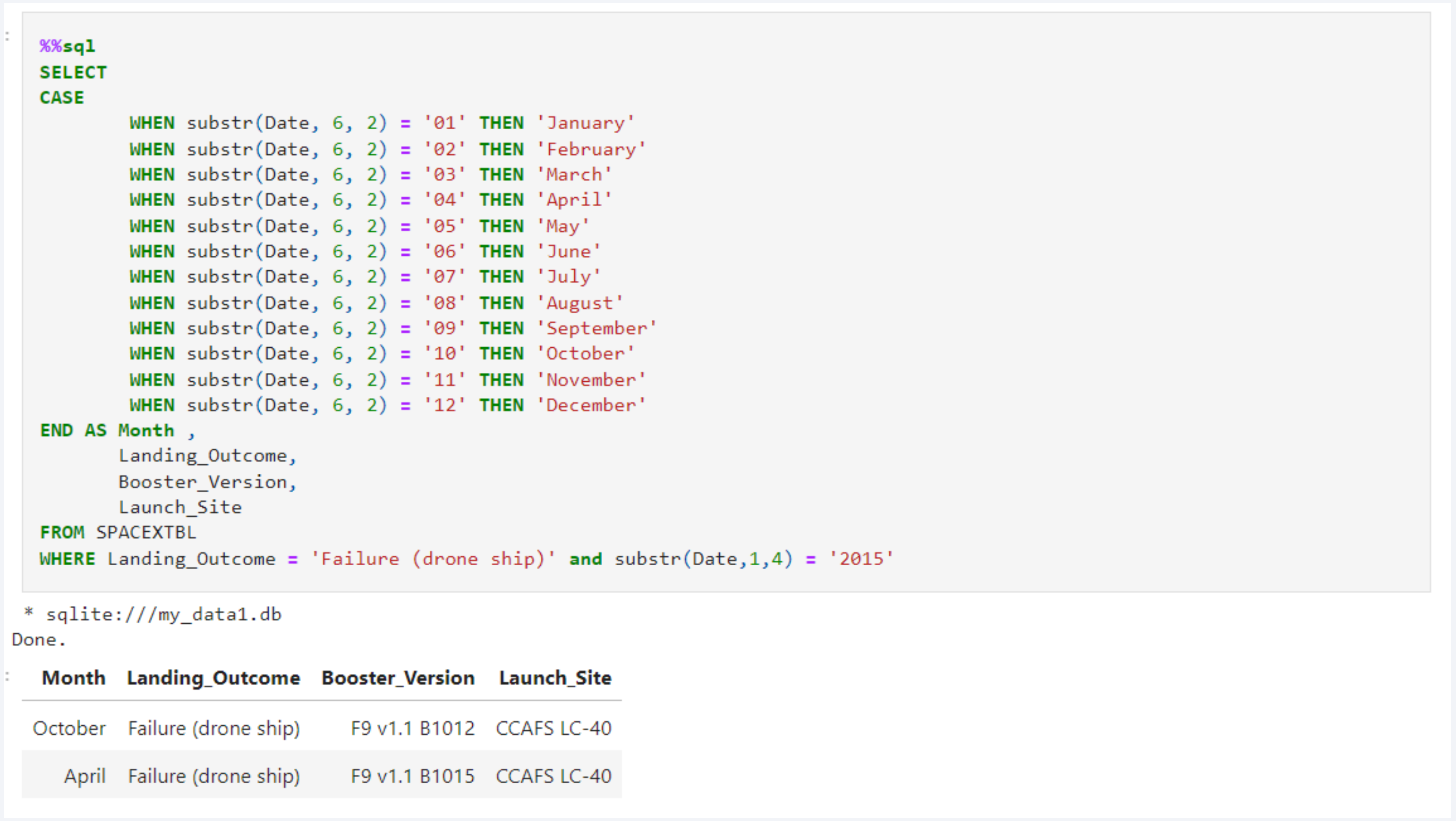
```
%sql select Booster_Version from SPACEXTBL where PAYLOAD_MASS__KG_ = ( select max(PAYLOAD_MASS__KG_) from SPACEXTBL)
```

```
 * 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

- We used the query below to list the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015.



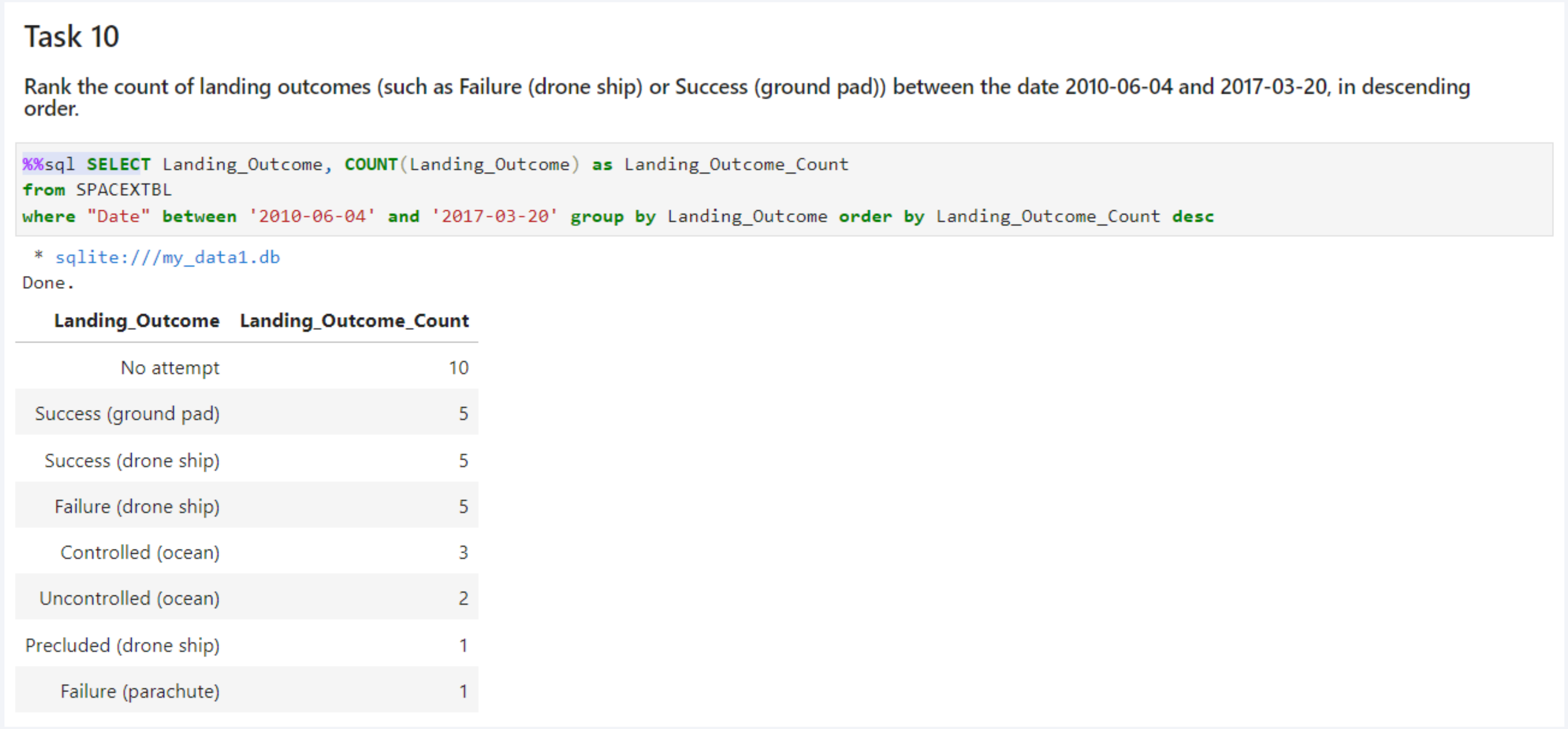
```
%%sql
SELECT
CASE
        WHEN substr(Date, 6, 2) = '01' THEN 'January'
        WHEN substr(Date, 6, 2) = '02' THEN 'February'
        WHEN substr(Date, 6, 2) = '03' THEN 'March'
        WHEN substr(Date, 6, 2) = '04' THEN 'April'
        WHEN substr(Date, 6, 2) = '05' THEN 'May'
        WHEN substr(Date, 6, 2) = '06' THEN 'June'
        WHEN substr(Date, 6, 2) = '07' THEN 'July'
        WHEN substr(Date, 6, 2) = '08' THEN 'August'
        WHEN substr(Date, 6, 2) = '09' THEN 'September'
        WHEN substr(Date, 6, 2) = '10' THEN 'October'
        WHEN substr(Date, 6, 2) = '11' THEN 'November'
        WHEN substr(Date, 6, 2) = '12' THEN 'December'
END AS Month ,
       Landing_Outcome,
       Booster_Version,
       Launch_Site
FROM SPACEXTBL
WHERE Landing_Outcome = 'Failure (drone ship)' and substr(Date,1,4) = '2015'
```

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

Month	Landing_Outcome	Booster_Version	Launch_Site
October	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
April	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

- We used the query in below to 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.



Task 10

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.

```
%%sql SELECT Landing_Outcome, COUNT(Landing_Outcome) as Landing_Outcome_Count
from SPACEXTBL
where "Date" between '2010-06-04' and '2017-03-20' group by Landing_Outcome order by Landing_Outcome_Count desc
```

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

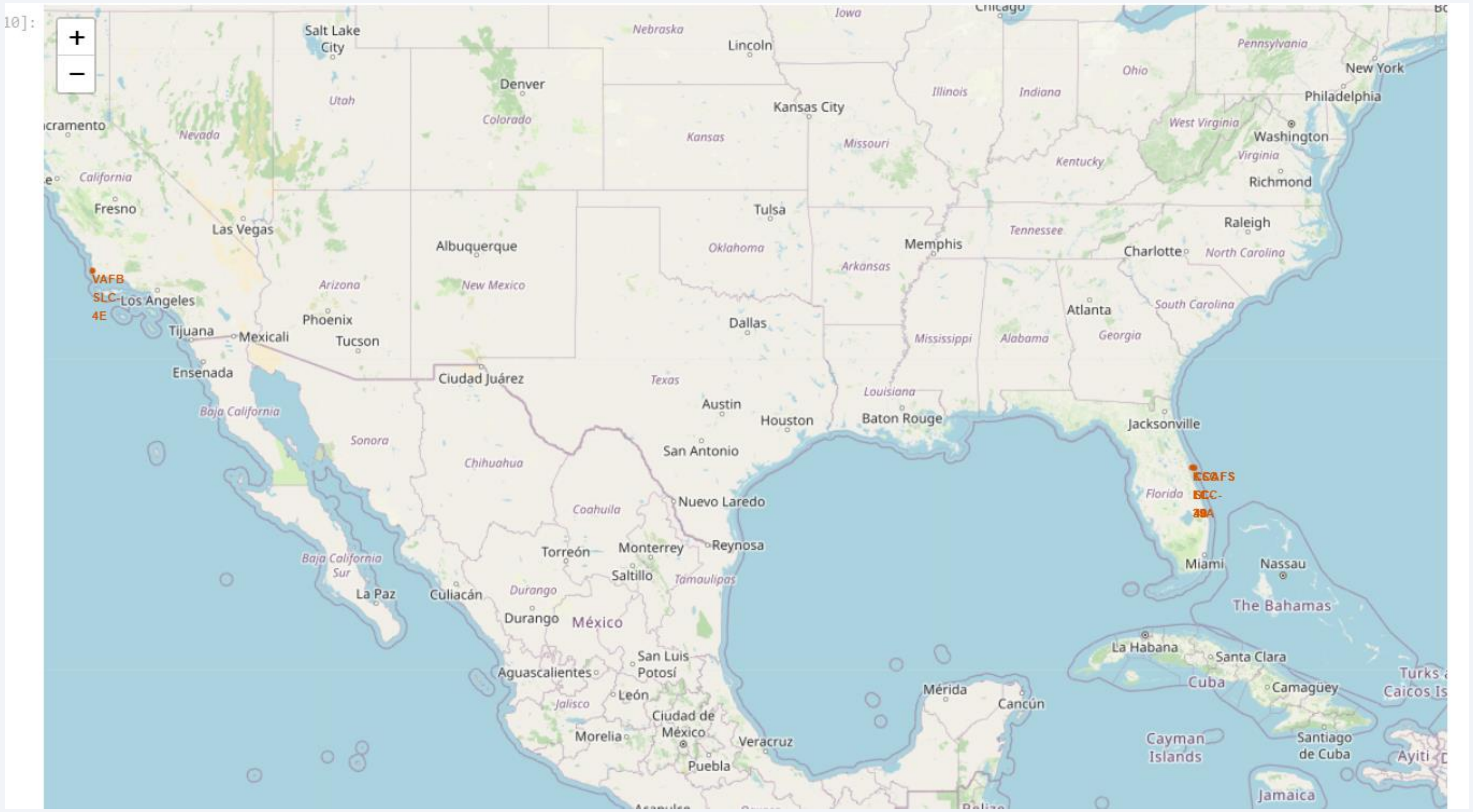
Landing_Outcome	Landing_Outcome_Count
No attempt	10
Success (ground pad)	5
Success (drone ship)	5
Failure (drone ship)	5
Controlled (ocean)	3
Uncontrolled (ocean)	2
Precluded (drone ship)	1
Failure (parachute)	1

Section 3

Launch Sites Proximities Analysis

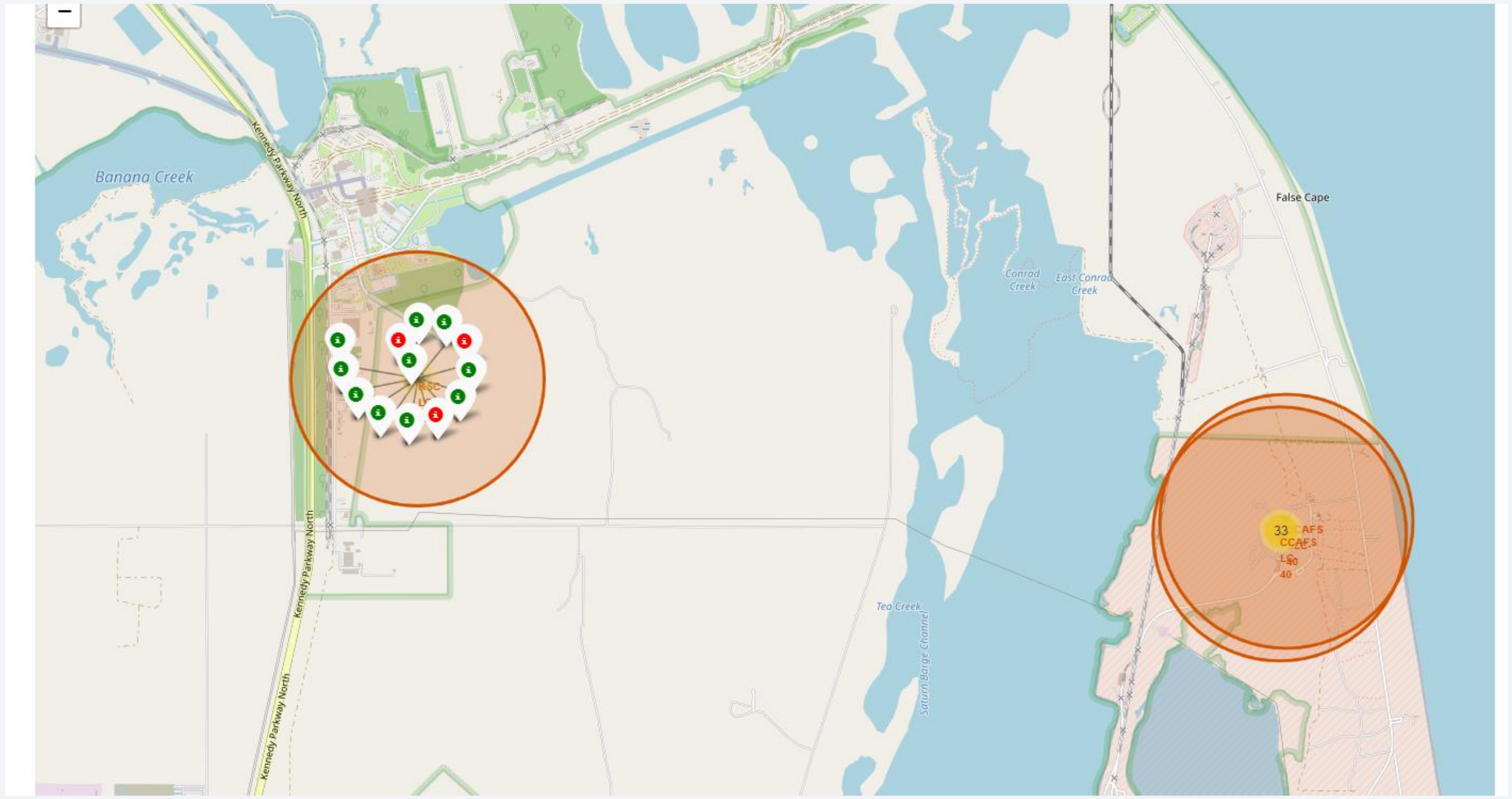
Launch Sites Analysis with Folium

- We created and added folium.Circle and folium.Marker for each launch site on the site map with using launch_sites dataframe. You can explore the code below.
- We can see te all launch sites are near a coast side, railway or a city and close to equator line.



Launch Outcomes Analysis Example with Folium

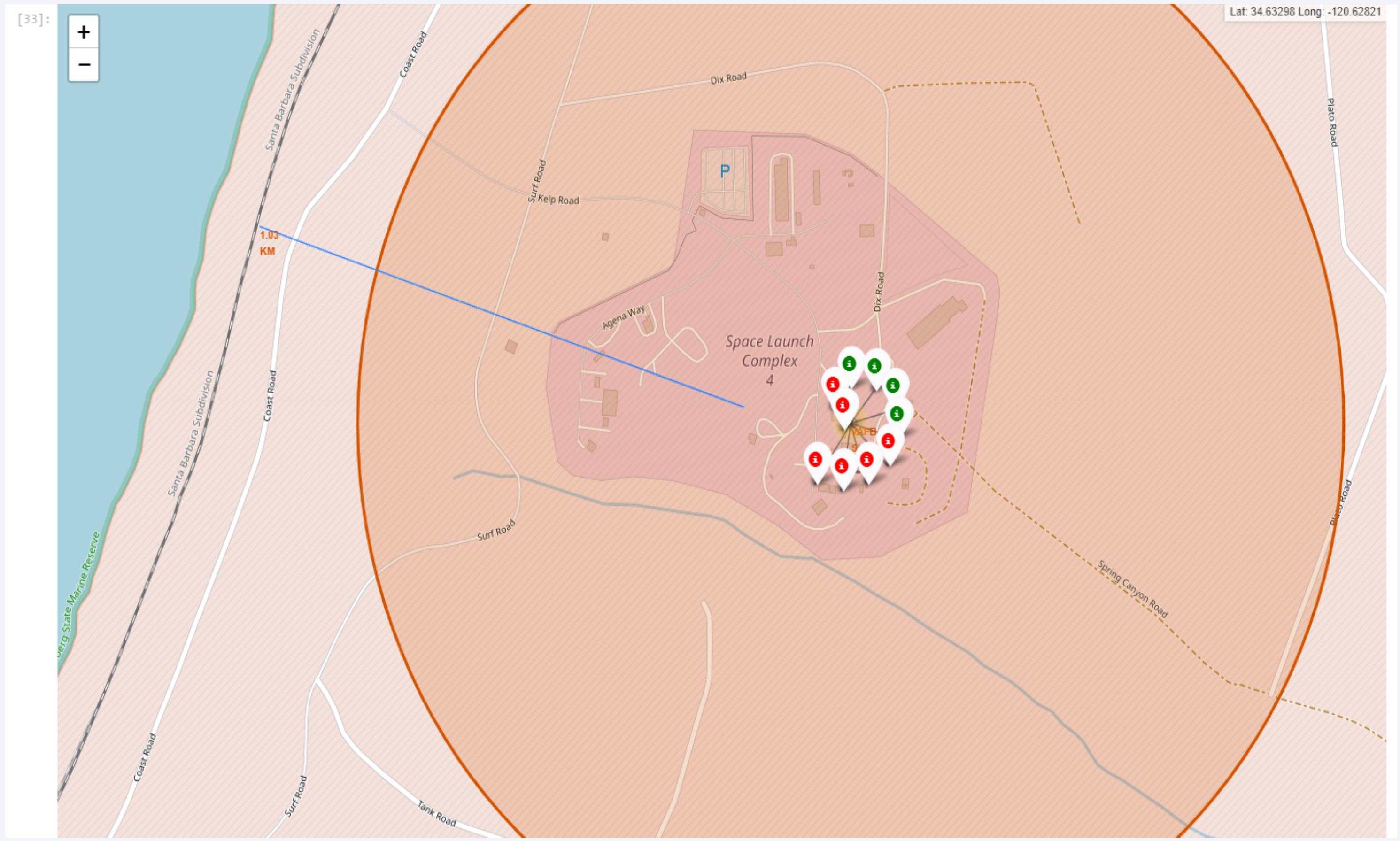
- Red markers represents failed launches, green markers represents successful launches.
- Kennedy Space Center Launch Complex 39A has relatively high success rate.



Vandenberg Space Launch Complex 4 Proximities with Santa Barbara Subdivision Railway

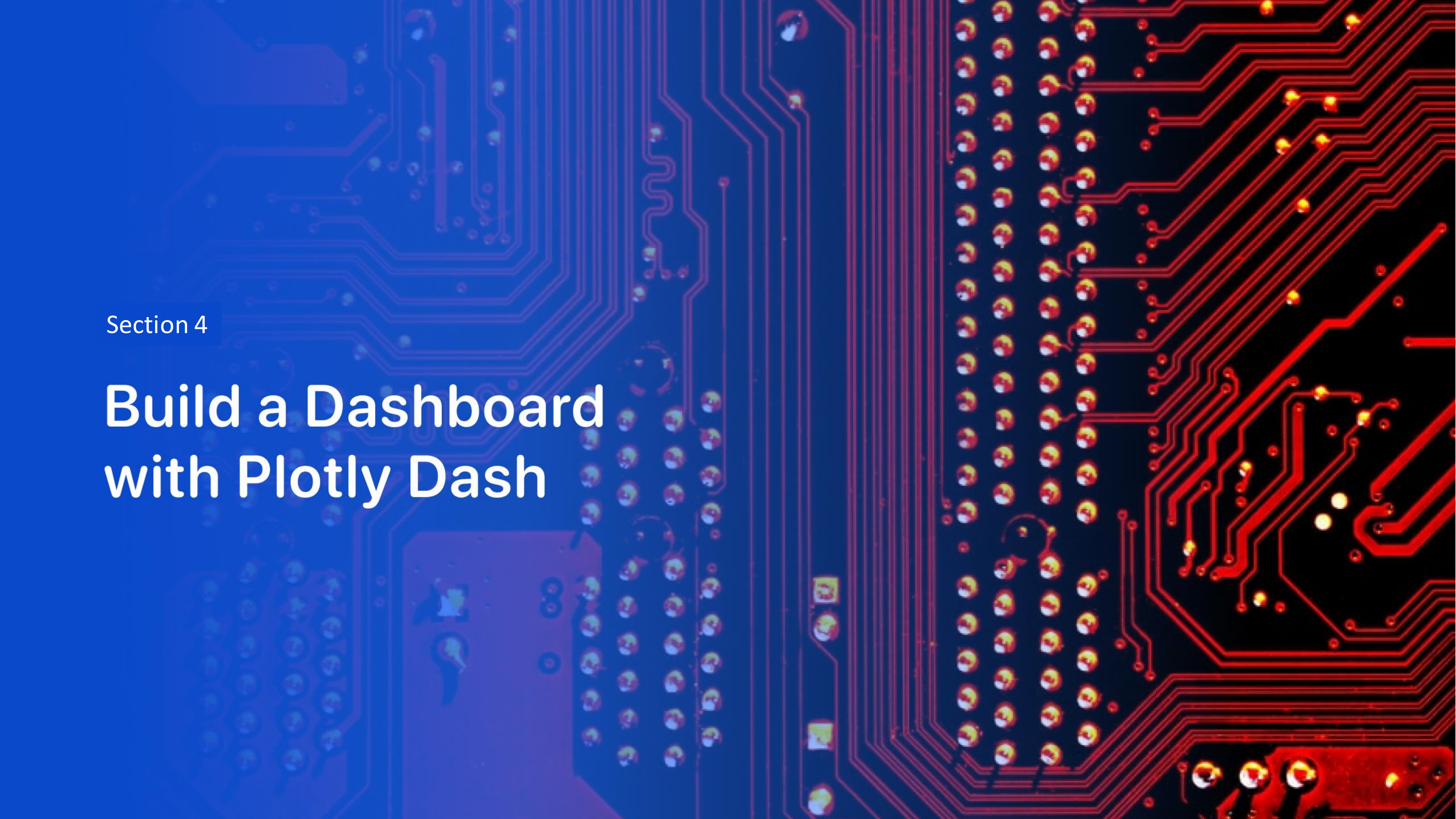
- We use a function for calculate the distance between VSLC 4E with Santa Barbara Subdivision Railway. Than plot a line between these two with 'PolyLine'.

- While the launch areas are very close to areas such as railways and the seashore. They were built away from settlements to ensure security.



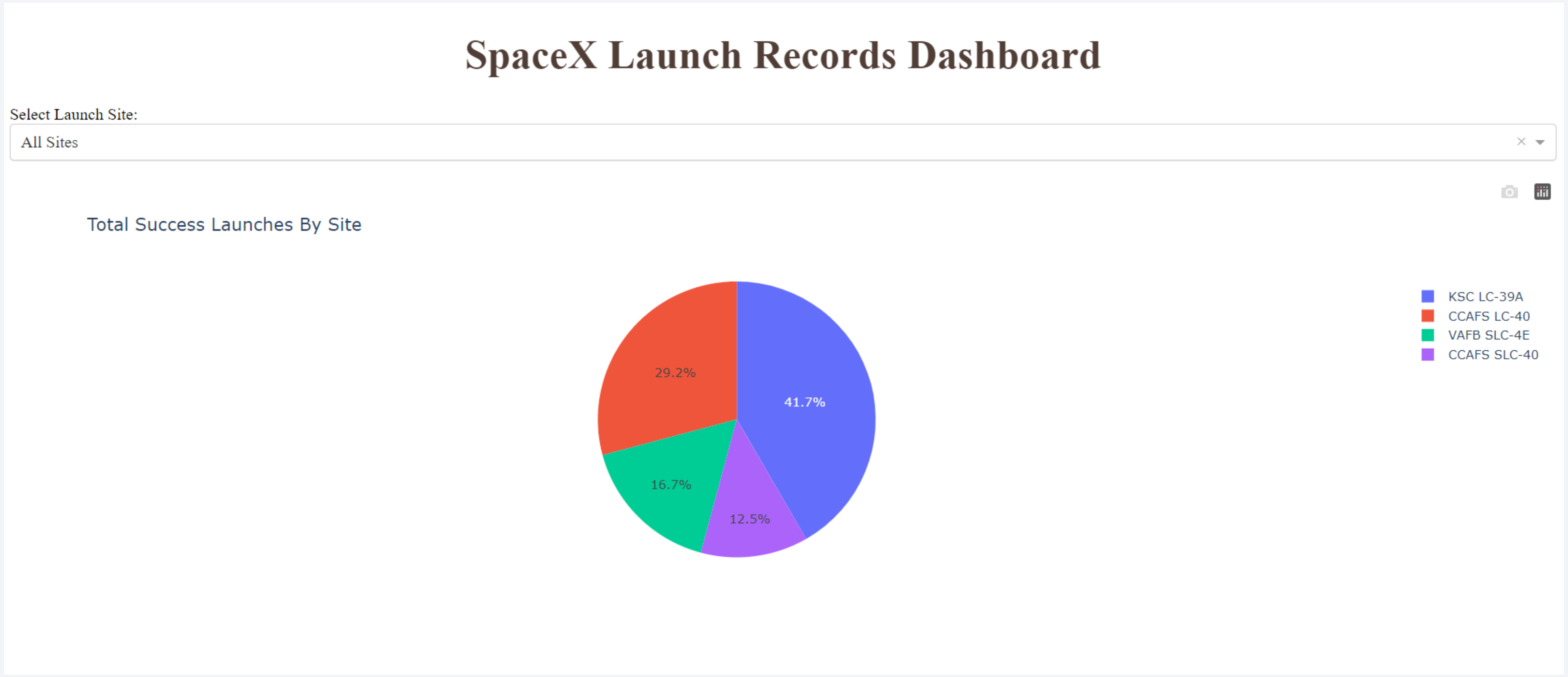
Section 4

Build a Dashboard with Plotly Dash



Dash Board Total Launch Success for all Sites

- As seen in the pie chart, KSC LC-39A is the launch site with the most successful landings.



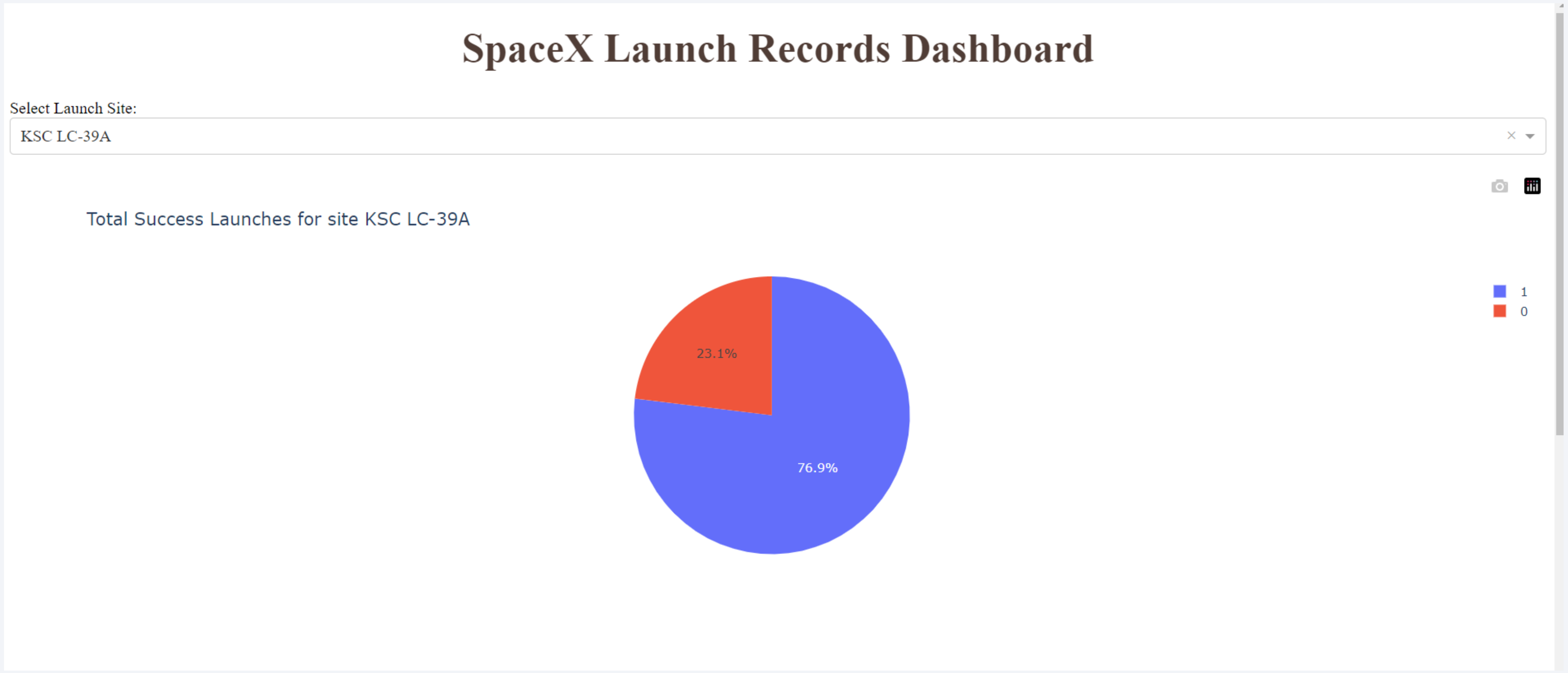
SpaceX Launch Records Dashboard

Select Launch Site:

All Sites

Dash Board Total Launch Success for each Sites

- When all launch sites are examined, KSC LC-39A is the launch site with the highest success rate.

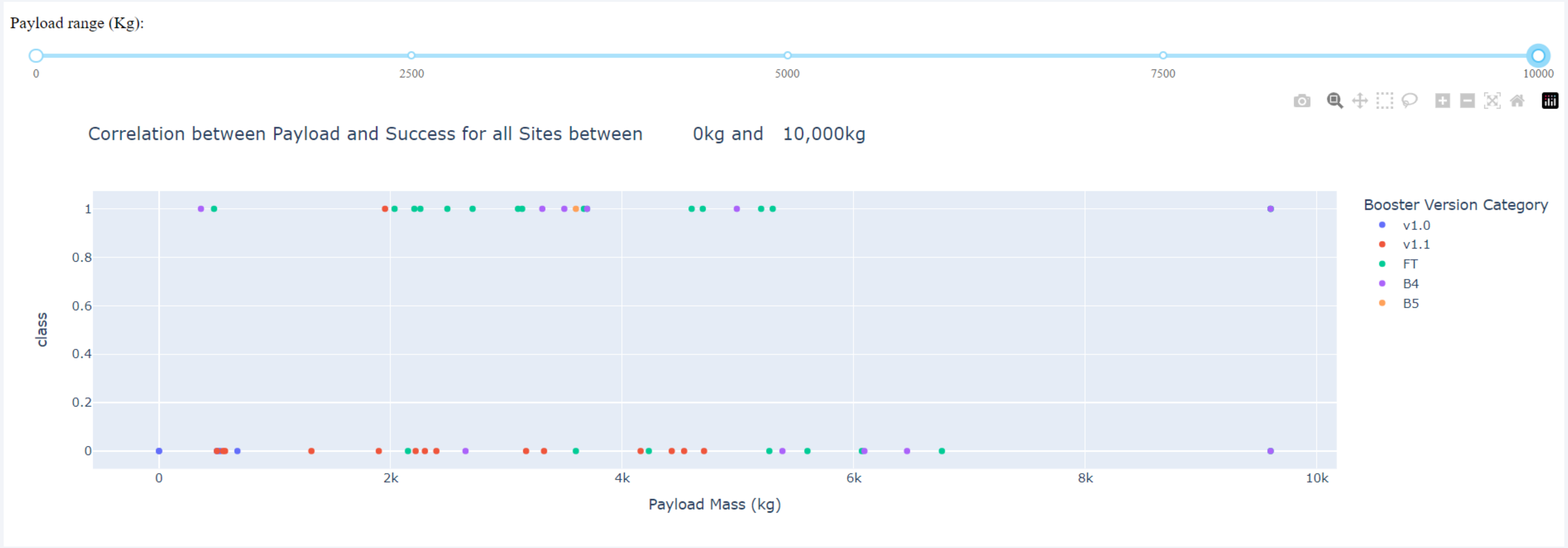

SpaceX Launch Records Dashboard

Select Launch Site:

KSC LC-39A

Dash Board Payload Mass vs Launch Outcome with Payload Mass Range Slider for all Sites

- The B5 booster version appears to have the highest success rate because we only have data from one trial. FT has the highest success rate among other booster versions.
- Payload mass between 2000 and 6000 kg has the highest success rate.
- Payload mass between 6000 and 10000 kg has the lowest success rate.

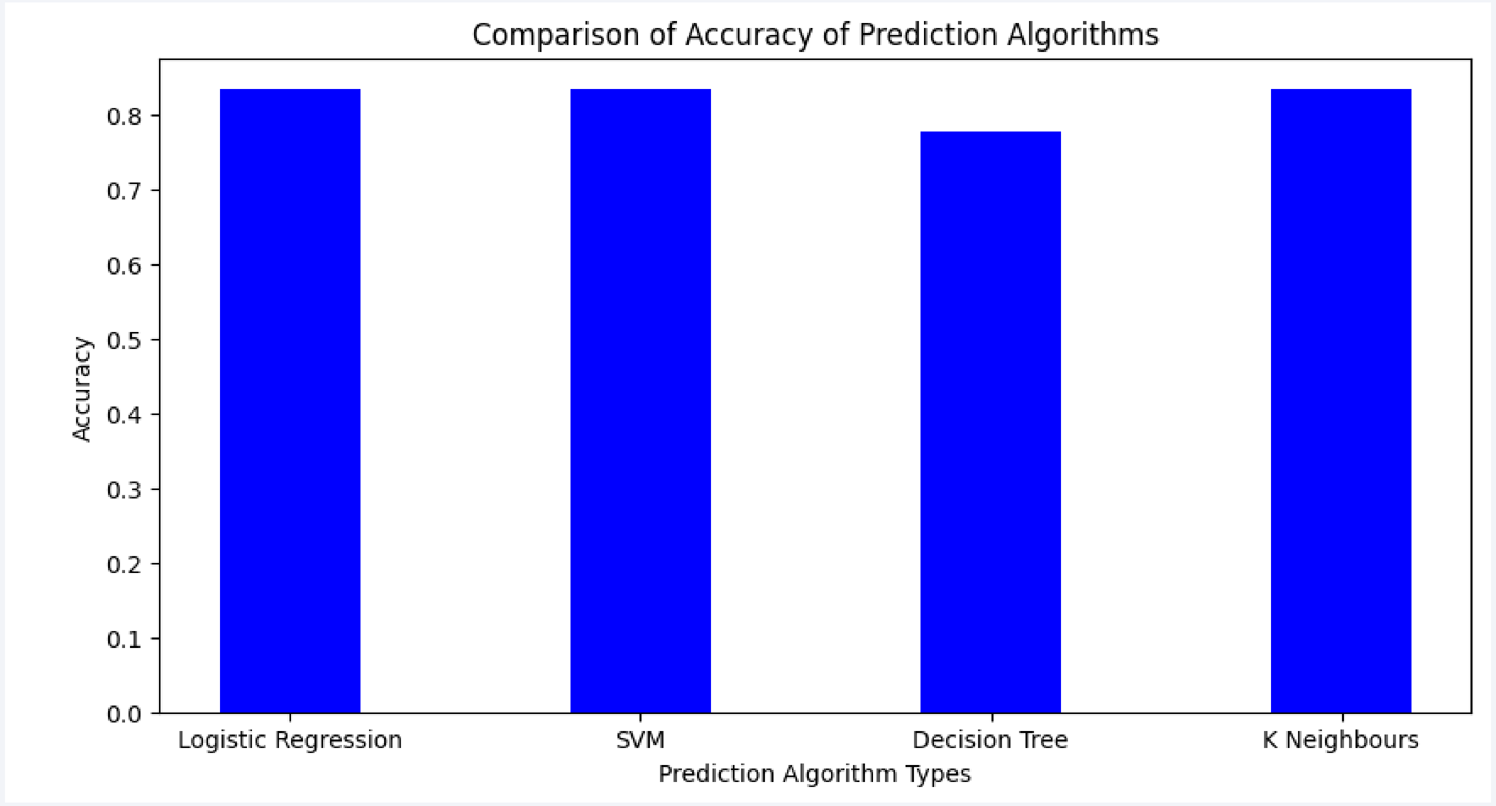


Section 5

Predictive Analysis (Classification)

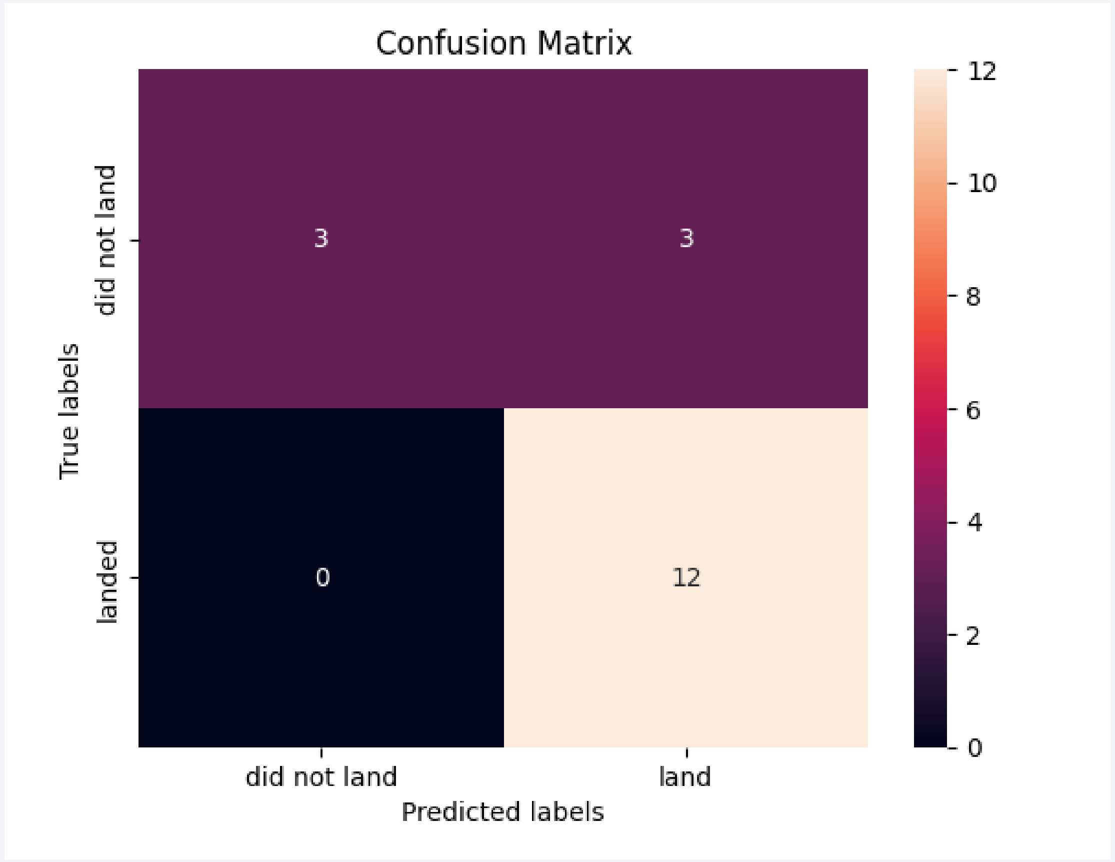
Classification Accuracy

- Visualize the built model accuracy for all built classification models, in a bar chart
- All classification algorithms have the same accuracy score "0.8333333333333334" except decision tree algorithm has "0.7777777777777778" accuracy score.



Confusion Matrix

- Logistic Regression, SVM and KNN Regression algorithms have same accuracy scores as shown at the below Confussion Matrixes:



Conclusions

- Launches with a payload mass over 6000 kg pose a risk.
- Kennedy Space Center Launch Complex 39A can be chosen as the most successful launch site.
- Among the decision-making algorithms, algorithms other than the decision tree method can be chosen as a prediction method and the characteristics of the launch can be selected under appropriate conditions.

Appendix

- The graphics in the presentation were prepared using Jupyter notebook.

Thank you!