IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Jón Helgi Elínar Kjartansson
November 20th 2024

Outline

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

Executive Summary

- Data was collected via SpaceX API and web scraping, cleaned through data wrangling, analyzed using EDA with visualizations, predictive models were built and evaluated, and interactive analytics were created using Folium and Plotly Dash.
- The analysis revealed key factors influencing SpaceX launch success, with payload mass and orbit type being significant predictors. Predictive models achieved high accuracy, and interactive visualizations showcased launch trends and success rates across various conditions.

Introduction

- This project analyzes SpaceX launch data to identify factors that influence launch success, using data science techniques to explore historical trends and build predictive models for future outcomes.
- Can we accurately predict the success of a SpaceX launch based on factors such as payload mass, orbit type, and booster version, and what are the key factors that most influence the likelihood of success?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected from the SpaceX API for launch details and through web scraping for additional historical information.
- Perform data wrangling
  - Data was cleaned by handling missing values, reformatting dates, converting categorical variables to numeric, and consolidating data from multiple sources into a structured format for analysis.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash

Methodology

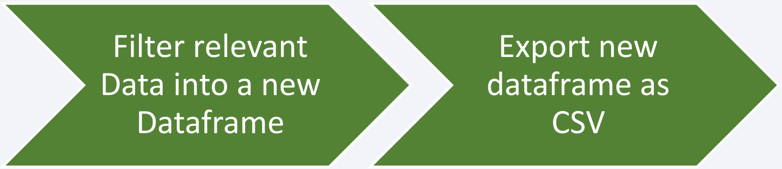
Executive Summary

- Perform predictive analysis using classification models
  - Classification models were built to predict launch success using features like payload mass, orbit type, and booster version. The models were tuned using hyperparameter optimization and evaluated based on accuracy, with confusion matrices used to assess performance and identify the best model.

Data Collection

- The primary dataset was collected using the SpaceX REST API, which provided detailed launch data including flight numbers, payload mass, launch site, orbit type, and success or failure outcomes. API calls retrieved this data in JSON format, which was then stored for further analysis. Additionally, BeautifulSoup was used for web scraping to gather supplementary historical data and details not available via the API, such as booster versions and launch site specifics. This ensured a comprehensive dataset combining structured API data with unstructured web-scraped information.

Data Collection

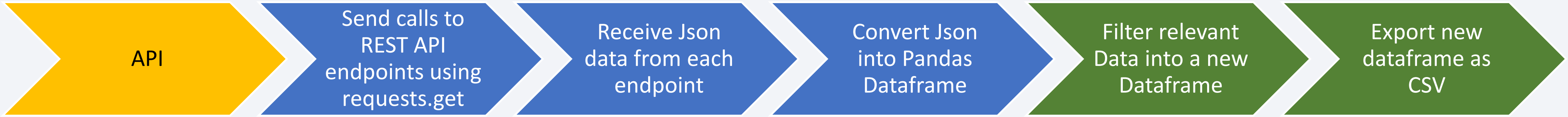


Filter relevant Data into a new Dataframe

Export new dataframe as CSV

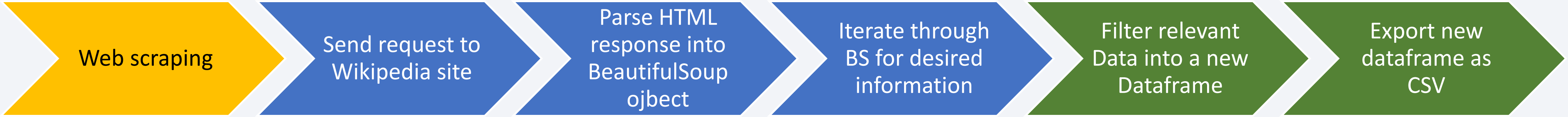
Data Collection – SpaceX API

- SpaceX offers an API that is well documented here. It has a number of endpoints that each returns different data, making it easy to collect the exact information desired without fetching a lot of irrelevant data. Here is the link to my notebook on Github.


API
Send calls to REST API endpoints using requests.get
Receive Json data from each endpoint
Convert Json into Pandas Dataframe
Filter relevant Data into a new Dataframe
Export new dataframe as CSV

Data Collection - Scraping

- The HTML for the whole Wikipedia Site for Falcon 9 and Falcon heavy launches was fetched and then turned into a BeautifulSoup object, making it possible to extract the tables in the code. The data from the tables was then put into a dataframe. Here is the link to my notebook on Github.



Web scraping

Send request to Wikipedia site

Parse HTML response into BeautifulSoup ojbect

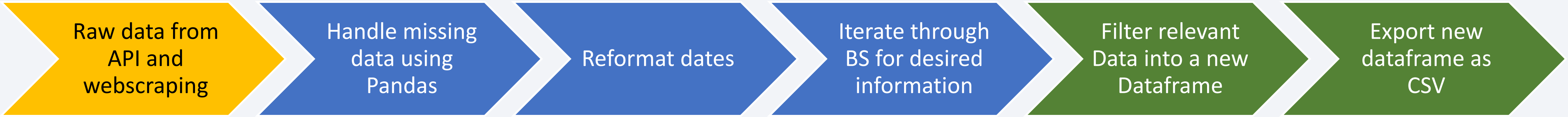
Iterate through BS for desired information

Filter relevant Data into a new Dataframe

Export new dataframe as CSV

Data Wrangling

- Data was processed by first handling missing values, then reformatting date fields, converting categorical variables to numeric, merging datasets from API and web scraping, and finally normalizing numerical data for consistent scaling. Here is the link to my notebook on Github.



Raw data from API and webscraping

Handle missing data using Pandas

Reformat dates

Iterate through BS for desired information

Filter relevant Data into a new Dataframe

Export new dataframe as CSV

EDA with Data Visualization

- Scatter plots were used to visualize the relation between a number of factors in one chart
    1. The payload mass, the launch outcome and the development of their relation over time
    2. The launch site, the launc outcome and the development of their relation over time
    3. The launch site, the payload mass and the outcome of the launch
    4. Orbit, the launch outcome and the development of their relation over time
    5. Orbit, the payload mass and the outcome of the launch
- Bar plot was used to visualize the outcome means of different orbits
- Line plot was used to visualize the launch outcomes over time
- Here is the link to my notebook on Github

EDA with SQL

- Display the names of the unique launch sites
- Display 5 records where the launch site begins with the string ‘CCA’
- Display the total payload mass carried by boosters launched by NASA (CRS)
- Display average payload mass carried by booster version F9 v1.1
- Return the date of the first successful ground pad landing
- Display the names of boosters with success in drone ship and 4000<payloadmass<6000
- List total number of successful and failed missions
- Display the names of boosters versions which carried the maximum payload mass
- Display mont, landing outcome, booster version and launch site for failed launchesin 2015
- Display all the landing outcomes with the count of each, in descending order
- Here is the link to my Github notebook

Build an Interactive Map with Folium

- **Objects that were put into the interactive Folium map**
  1. **Markers:**
     - Launch sites marked with names for easy identification.
     - Success/failed launch markers (green = success, red = failure) to visualize launch outcomes.
  2. **Circles:**
     - Highlight launch site locations and their surrounding area with radius-based circles.
  3. **Marker Clusters:**
     - Grouped launch markers to reduce clutter and improve map readability.
  4. **Distance Markers & Polylines:**
     - Distance lines between launch sites and nearby features (e.g., coastlines, cities) for proximity analysis.
  5. **MousePosition:**
     - Displays real-time coordinates for map exploration and distance calculation.
- **Purpose:**
  - **Visualize launch sites** and outcomes.
  - **Analyze geographic proximity** to key features (coastlines, cities).
  - **Identify trends** in launch success based on location.
- Here is the link to my notebook on Github

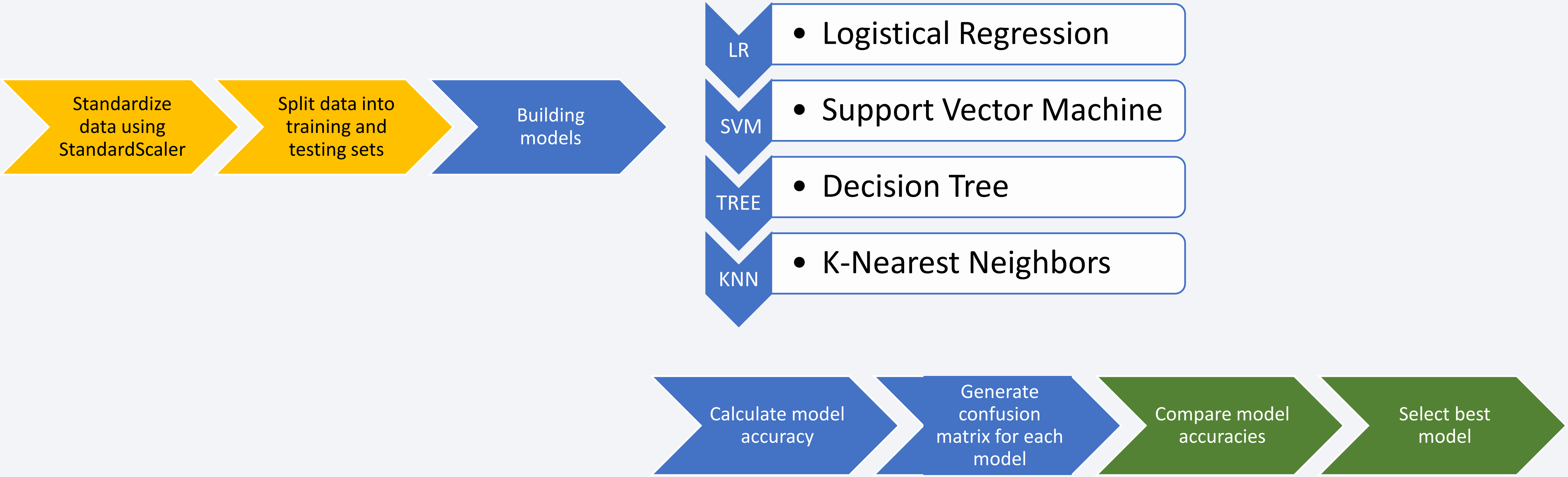
Build a Dashboard with Plotly Dash

- **Key Plots/Graphs:**
  1. **Pie Chart**:
     1. Displays success rates for all launch sites or success vs. failure for a selected site.
     2. *Purpose*: Compare launch performance across sites and evaluate success rates.
  2. **Scatter Plot**:
     1. Shows the relationship between payload mass and launch success, color-coded by booster version.
     2. *Purpose*: Explore how payload size impacts the outcome of launches.
- **Key Interactions:**
  1. **Launch Site Dropdown**:
     1. Select a specific launch site or view data for all sites, updating both the pie chart and scatter plot.
  2. **Payload Range Slider**:
     1. Adjust the payload range to filter the scatter plot data and examine how payload affects success.
- **Overall Purpose**: Provides interactive insights into SpaceX launch success rates, payload effects, and site-specific trends.
- Here is my python file including the Dash app on Github

Predictive Analysis (Classification)

1. Data Preprocessing
   - Standardize data with StandardScaler
   - Split data (80% testing, 20% training)
2. Model building (GridSearchCV)
   - **Logistic Regression**: Tuned C, penalty, solver.
   - **SVM:** Tuned kernel, C and gamma
   - **Decision Tree:** Tuned criterion, max_depth and splitter
   - **KNN:** Tuned n_neighbors, p and algorithm
3. Evaluation
   - Calculated accuracy for each model
   - Visualized performances using confusion matrices
4. Best model
   - Support Vector Machine (SVM) with 93.75% accuracy.

Predictive Analysis (Classification)



Standardize data using StandardScaler

Split data into training and testing sets

Building models

LR
- Logistical Regression

SVM
- Support Vector Machine

TREE
- Decision Tree

KNN
- K-Nearest Neighbors

Calculate model accuracy

Generate confusion matrix for each model

Compare model accuracies

Select best model

- Here is a link to my notebook on Github

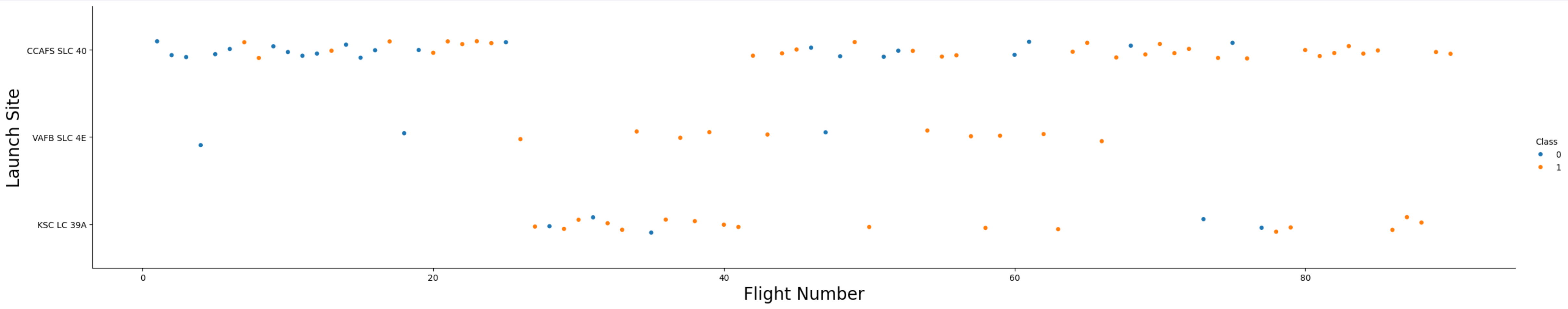
Results

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

Section 2

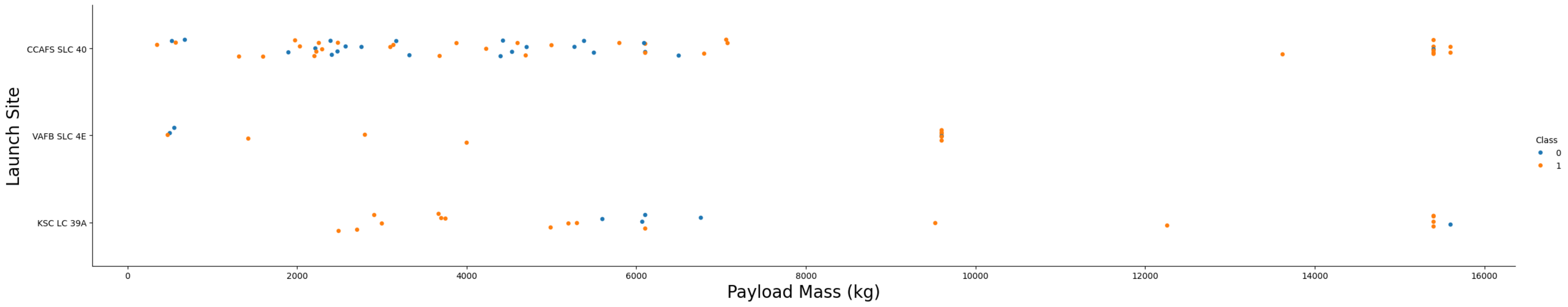
Insights drawn from EDA

Flight Number vs. Launch Site



- A scatter plot showing Launch Sites on the Y axis and Flight numbers on the X axis.
- The classification 0 or 1 implicates whether the landing was successful or not

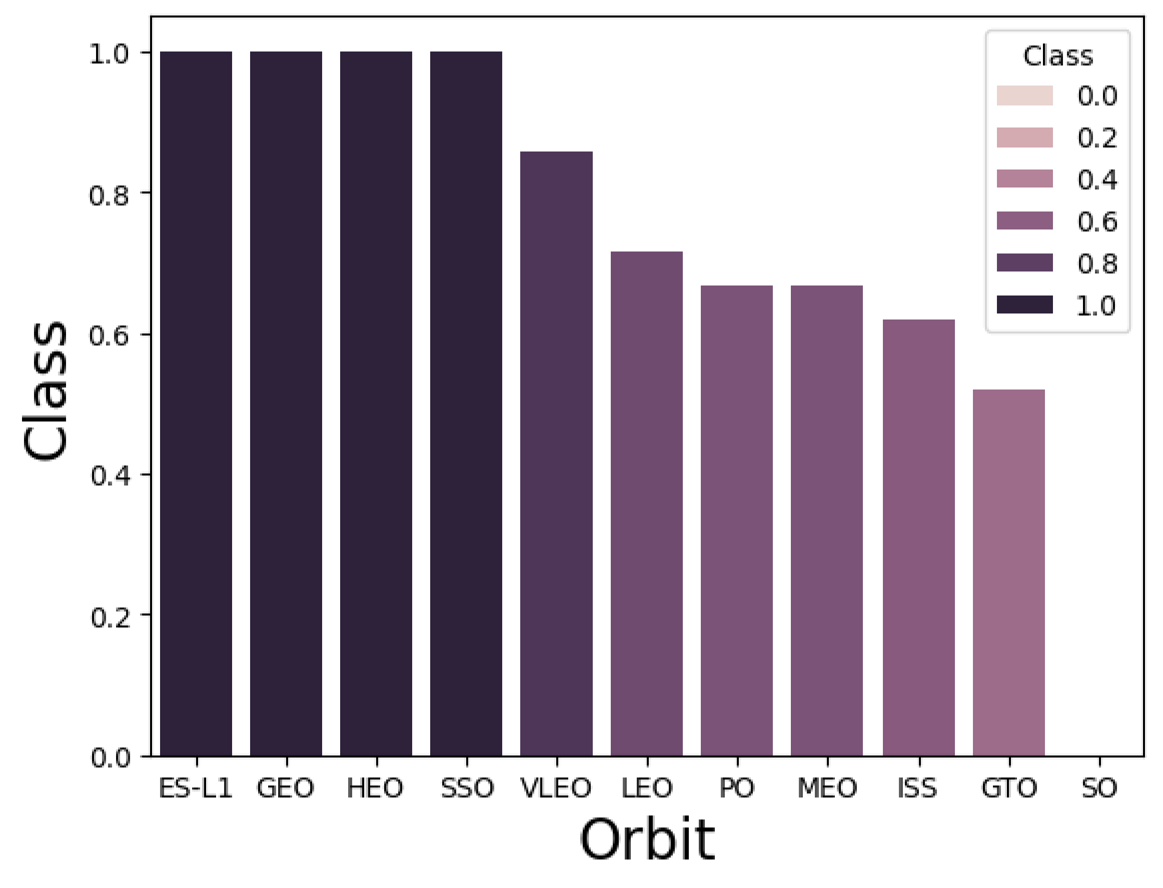
Payload vs. Launch Site



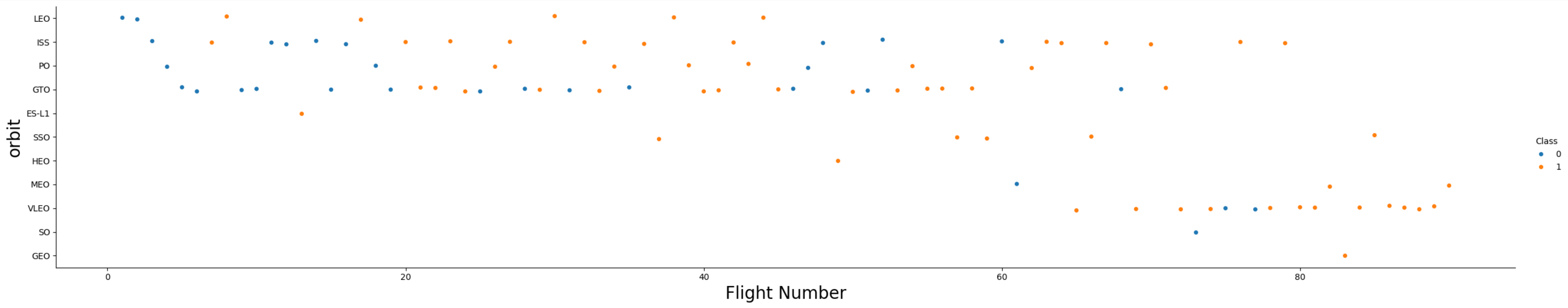
- A scatter plot with the launch sites on the Y axis and the Payload Mass on the X axis.
- The classification of 0 or 1 implicates whether the landing was successful or not.

Success Rate vs. Orbit Type

- This bar chart shows the median Class for each orbit type.
- We can therefor determine that the four orbit types that have the Class median 1.0 have never had a failed landing

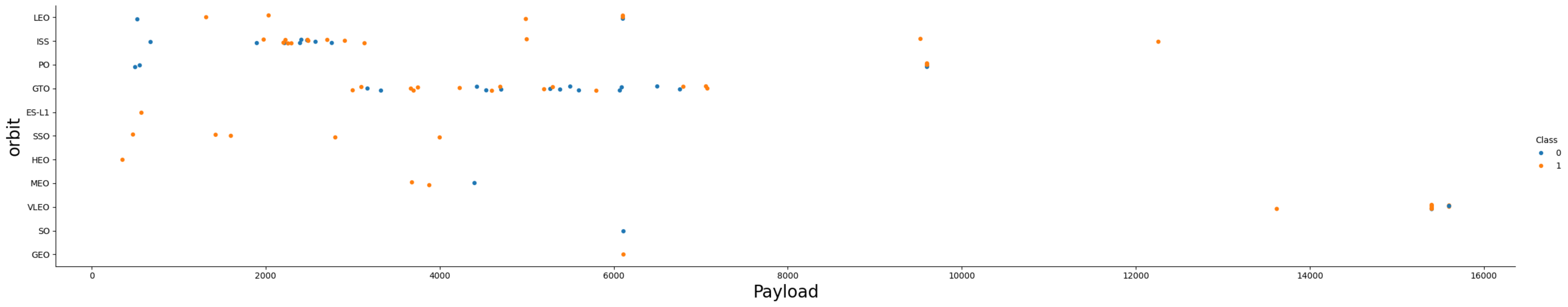


Flight Number vs. Orbit Type



- A scatter plot that has the Orbit type on the Y axis and the flight number on the X axis
- From this we can read how the orbit types have evolved over time
- The Classification of 0 and 1 also tells us the success of the landings of each launch

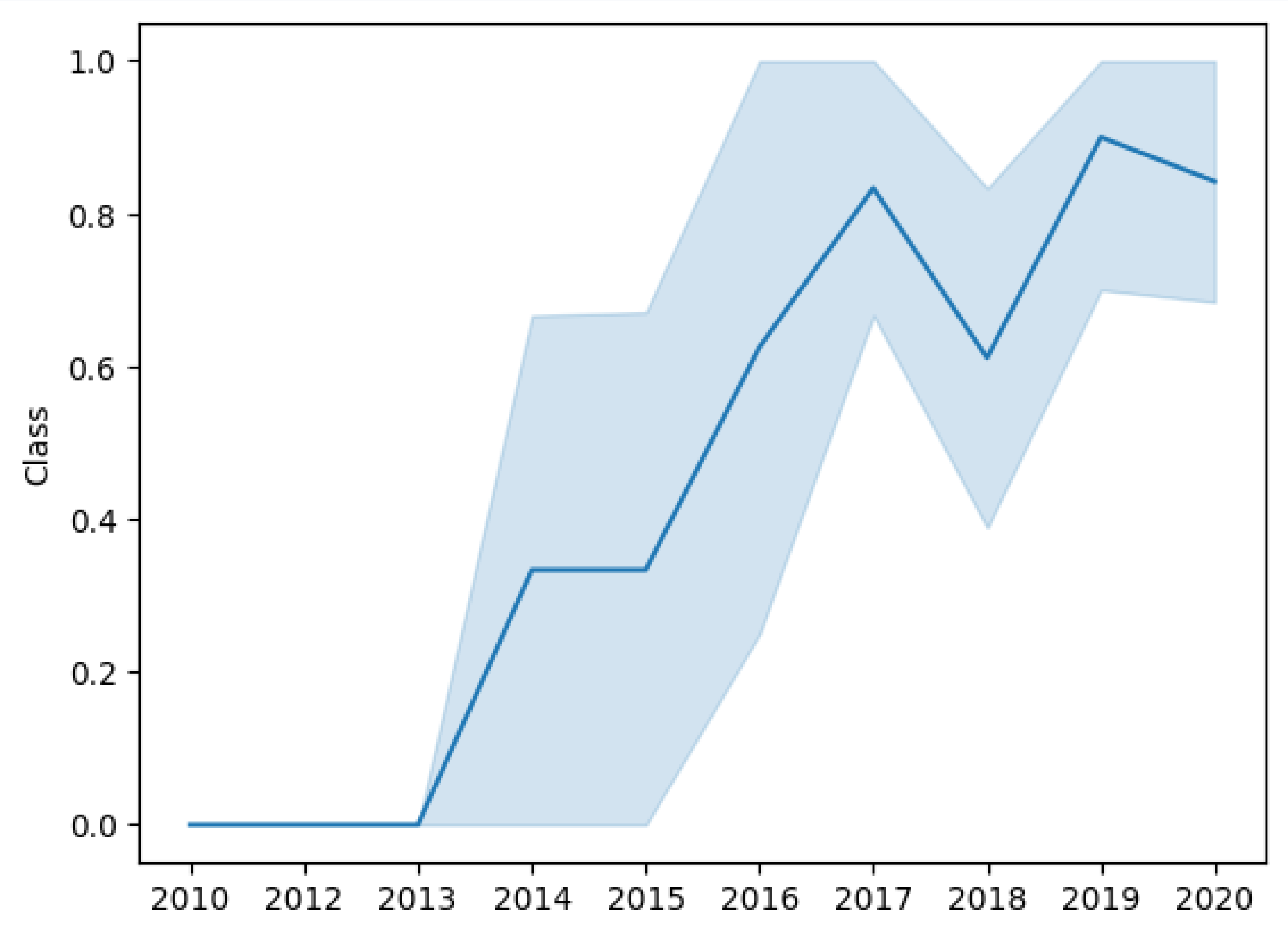
Payload vs. Orbit Type



- A scatter plot with the orbit types on the Y axis and the Payload Mass on the X axis
- The Classification of 0 and 1 for each launch tells us the success of the landing for that launch

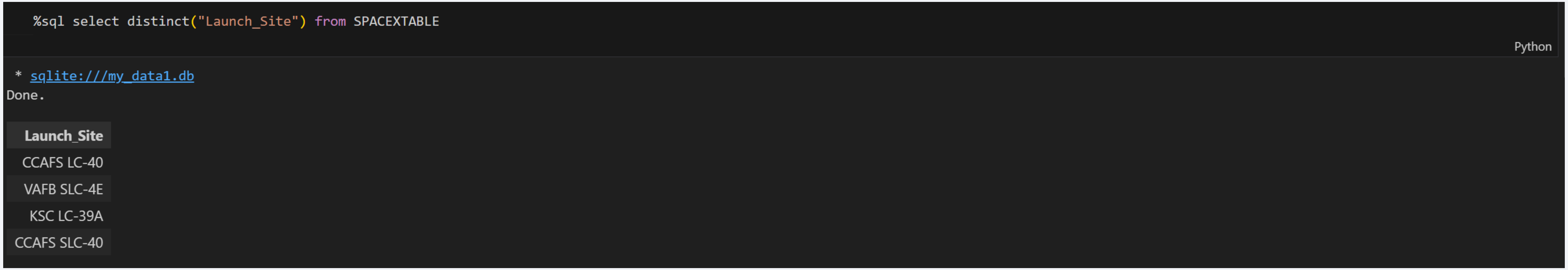
Launch Success Yearly Trend

- A line plot of the yearly average success rate



All Launch Site Names

- A simple query that returns all the unique Launch Site Names in a single column



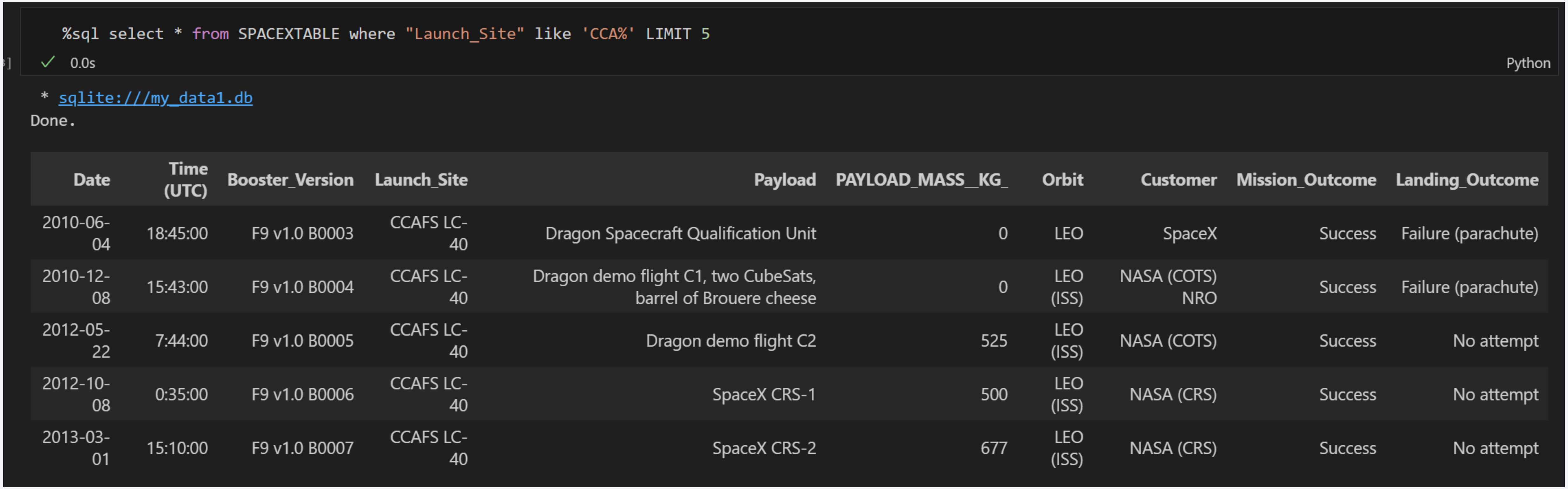
```
%sql select distinct("Launch_Site") from SPACEXTABLE
```

```
 * 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'

- A query returning the first 5 records that have a Launch Site Name beginning with ‘CCA’



```
%sql select * from SPACEXTABLE where "Launch_Site" like 'CCA%' LIMIT 5
```

✓ 0.0s

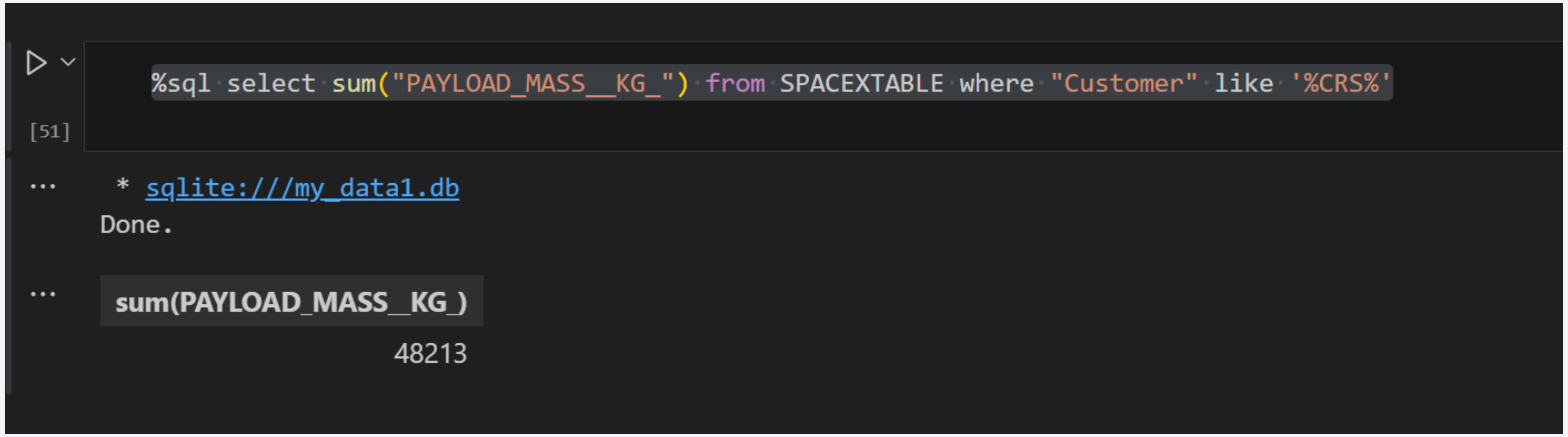
Python

```
 * 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

Total Payload Mass

- A query that Calculates the total payload carried by boosters from NASA using the sql SUM function



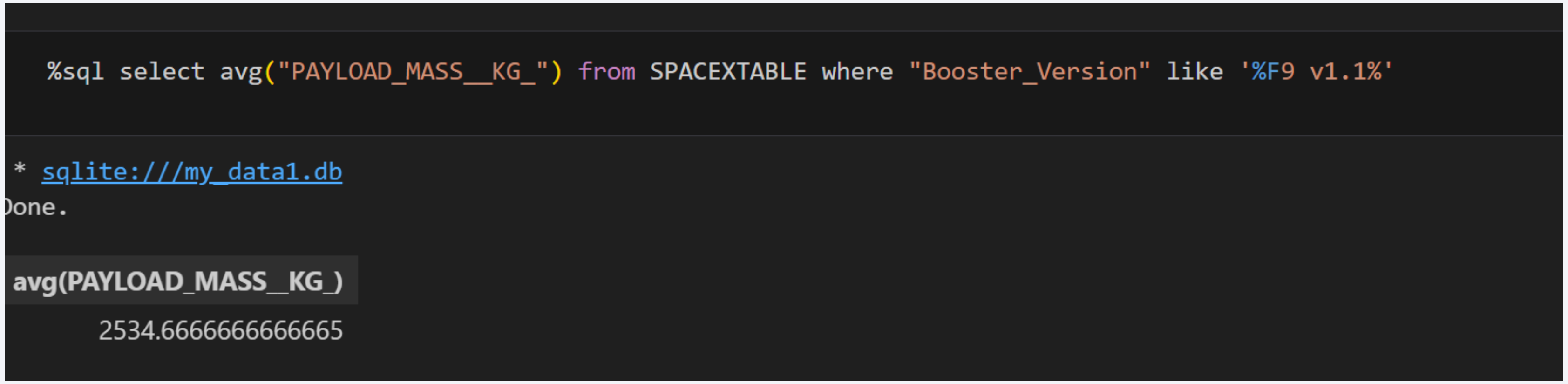
```
[51]
%sql select sum("PAYLOAD_MASS__KG_") from SPACEXTABLE where "Customer" like '%CRS%'

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

sum(PAYLOAD_MASS__KG_)
48213

Average Payload Mass by F9 v1.1

- A query that returns the average payload mass carried by booster version F9 v1.1 using the AVG function.



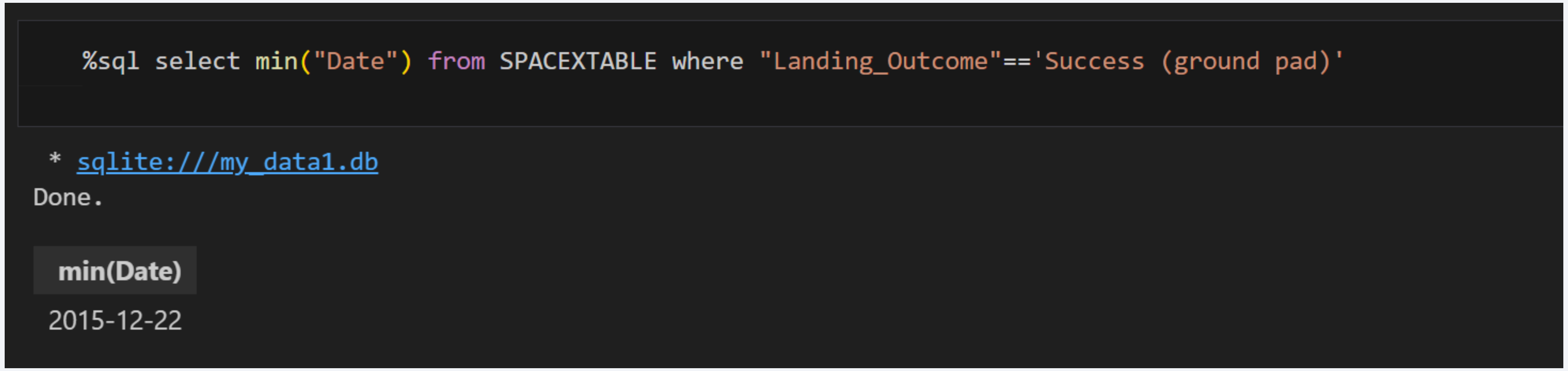
```
%sql select avg("PAYLOAD_MASS__KG_") from SPACEXTABLE where "Booster_Version" like '%F9 v1.1%'
```

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

avg(PAYLOAD_MASS__KG_)
2534.6666666666665

First Successful Ground Landing Date

- A query returning the date of the first successful landing outcome on ground pad by using the MIN function



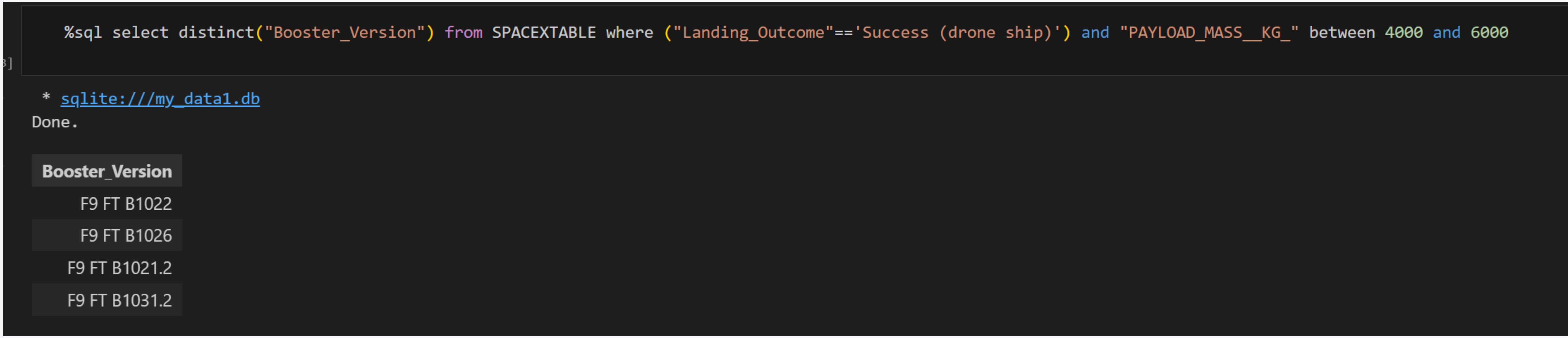
```
%sql select min("Date") from SPACEXTABLE where "Landing_Outcome"=='Success (ground pad)'
```

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

min(Date)
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- A query that lists the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000



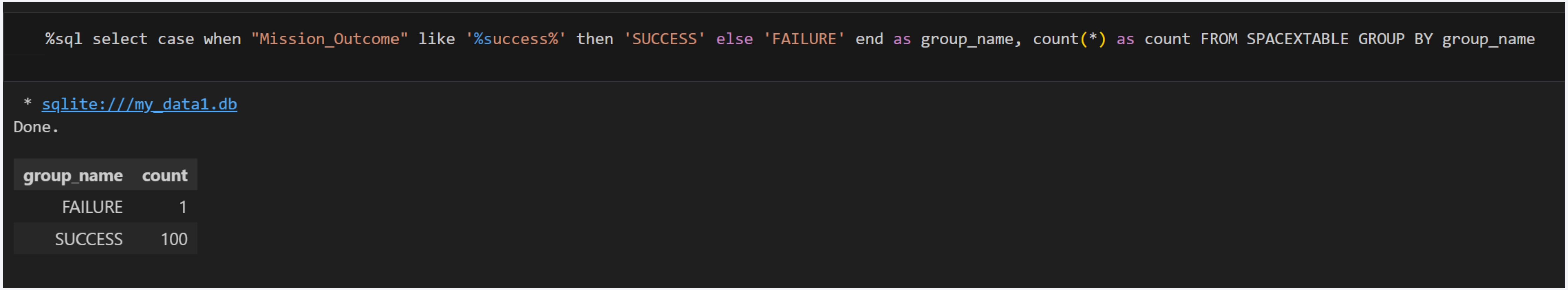
```
%sql select distinct("Booster_Version") from SPACEXTABLE where ("Landing_Outcome"=='Success (drone ship)') and "PAYLOAD_MASS__KG_" between 4000 and 6000
```

```
 * 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

- A function that Calculates the total number of successful and failure mission outcomes and outputs the results in a 2x2 table where one column includes the group name and one contains the number of missions

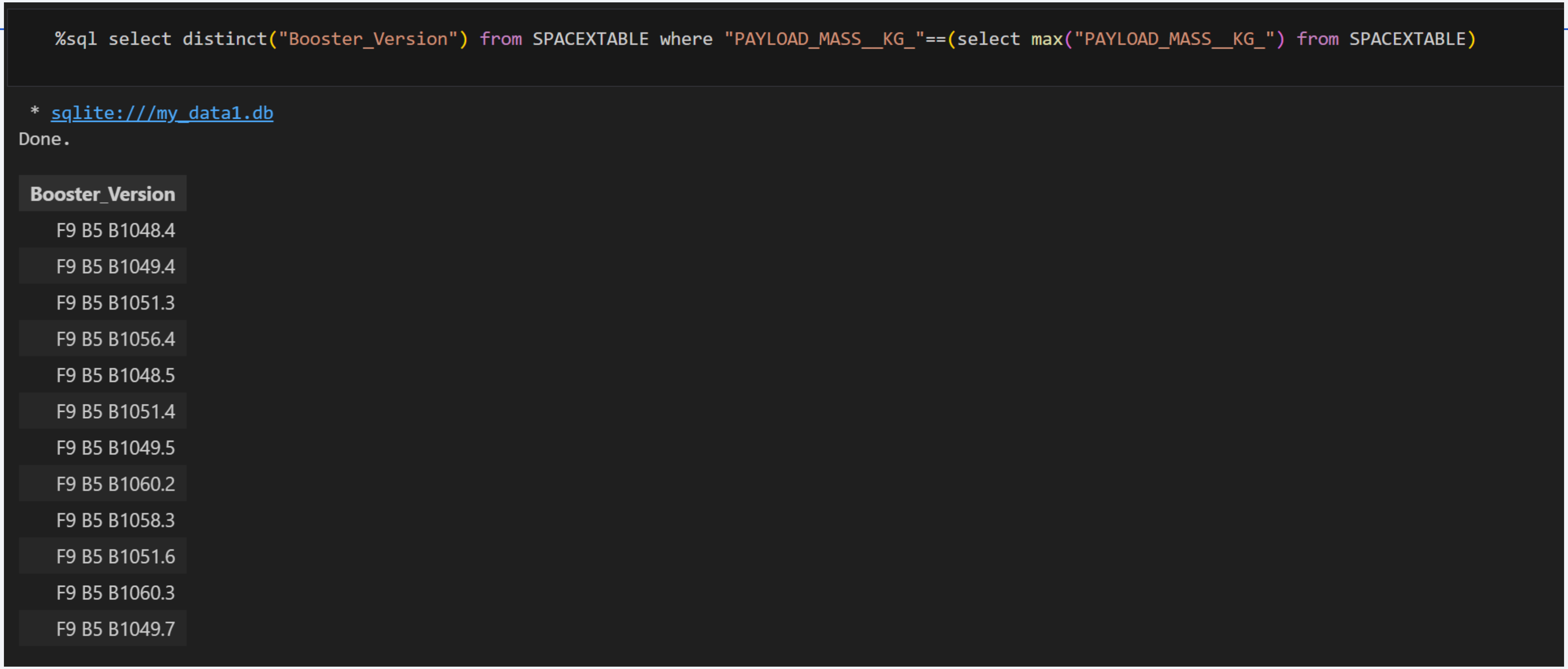


```
%sql select case when "Mission_Outcome" like '%success%' then 'SUCCESS' else 'FAILURE' end as group_name, count(*) as count FROM SPACEXTABLE GROUP BY group_name
```

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

group_name	count
FAILURE	1
SUCCESS	100

Boosters Carried Maximum Payload



```
%sql select distinct("Booster_Version") from SPACEXTABLE where "PAYLOAD_MASS__KG_"==(select max("PAYLOAD_MASS__KG_") from SPACEXTABLE)
```

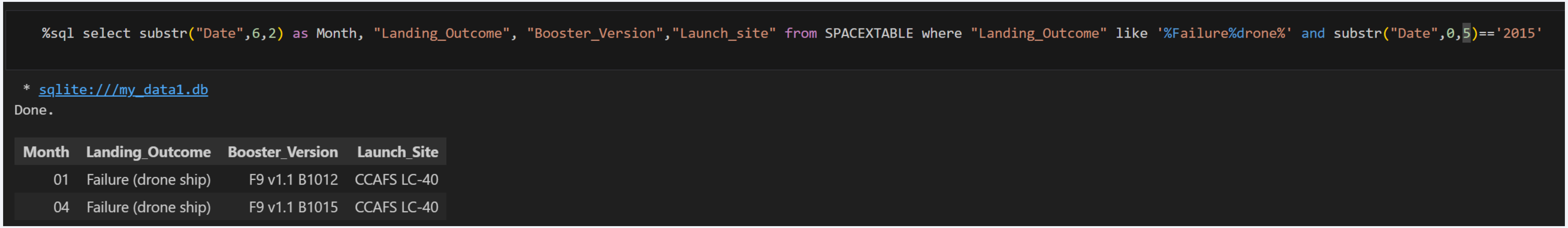
```
 * sqlite:///my_data1.db
Done.
```

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

- A query that returns a list of the names of the boosters which have carried the maximum payload mass

2015 Launch Records

- A query that lists the failed landing_outcomes in drone ship, their booster versions, month of launch and launch site names in the year 2015

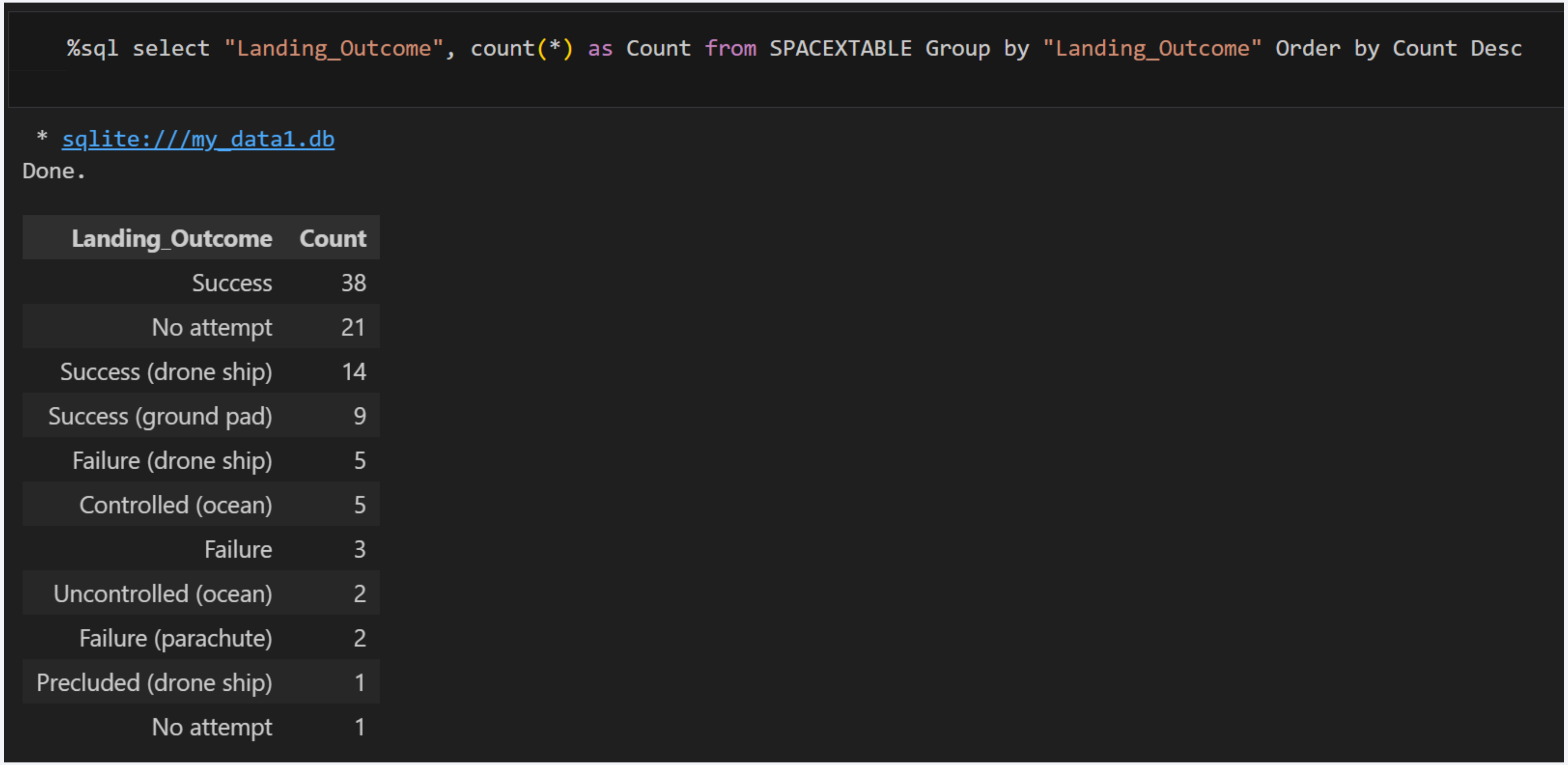


```
%sql select substr("Date",6,2) as Month, "Landing_Outcome", "Booster_Version","Launch_site" from SPACEXTABLE where "Landing_Outcome" like '%Failure%drone%' and substr("Date",0,5)=='2015'
```

```
 * 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 from SPACEXTABLE Group by "Landing_Outcome" Order by Count Desc
```

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

Landing_Outcome	Count
Success	38
No attempt	21
Success (drone ship)	14
Success (ground pad)	9
Failure (drone ship)	5
Controlled (ocean)	5
Failure	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1
No attempt	1

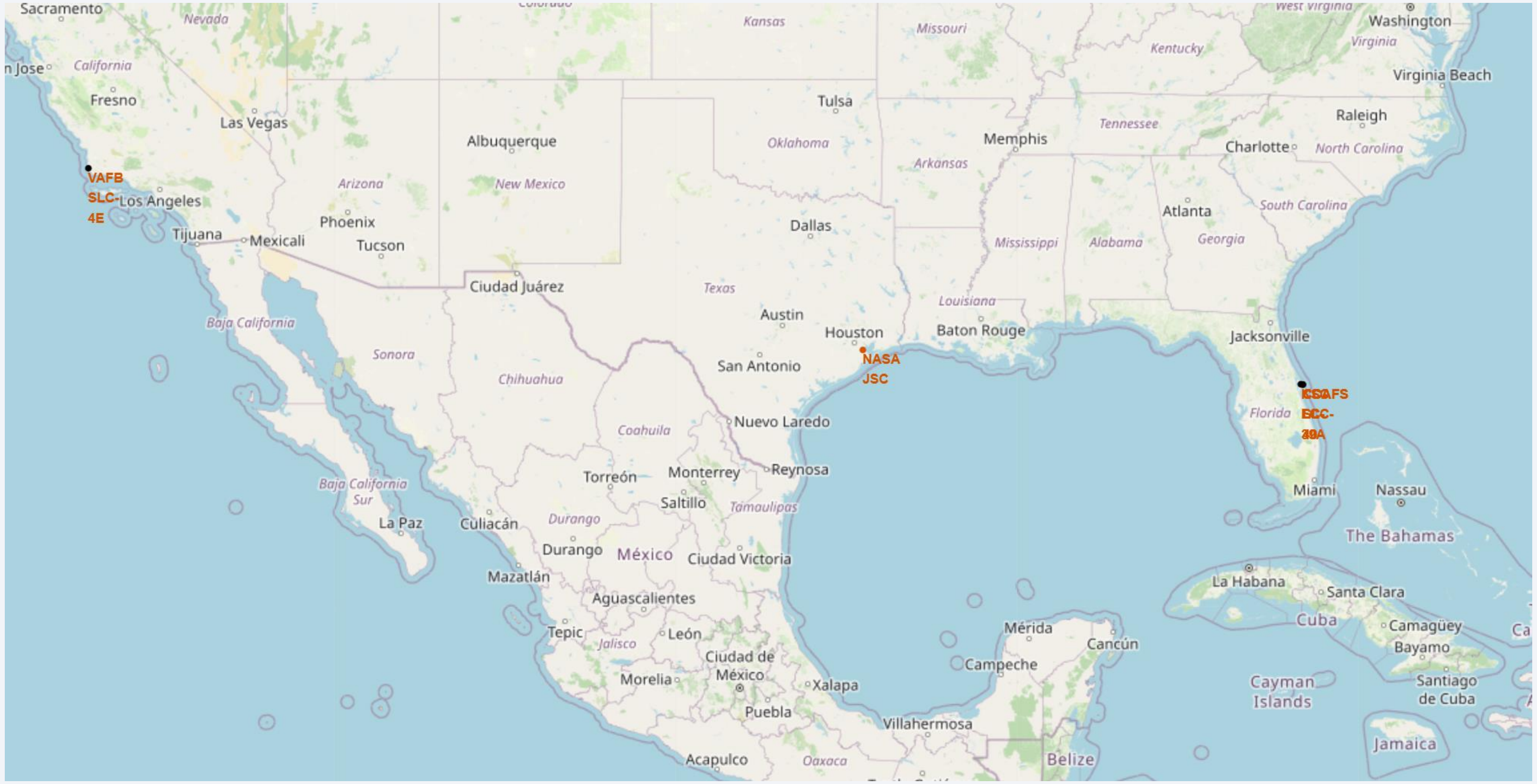
- A query that ranks the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order

Section 3

Launch Sites Proximities Analysis

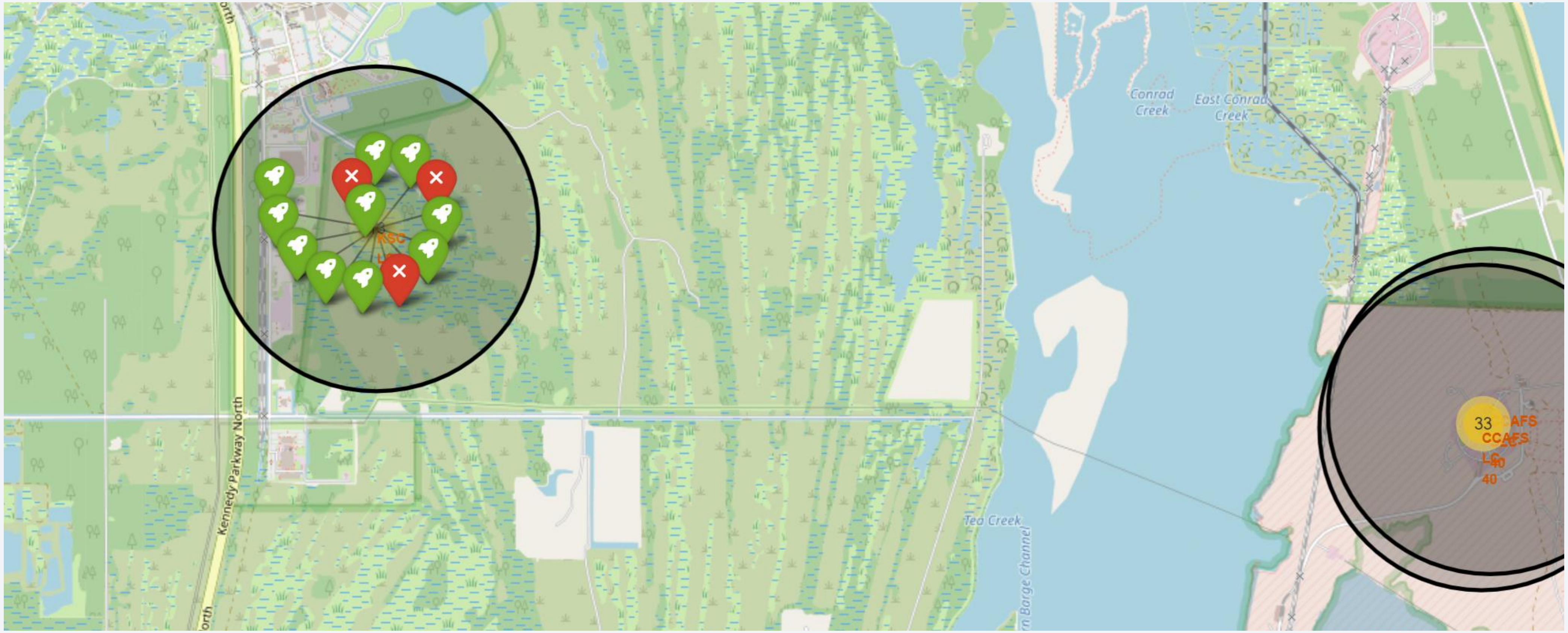
Launch site locations

- A folium map that includes markers for all the Launch sites



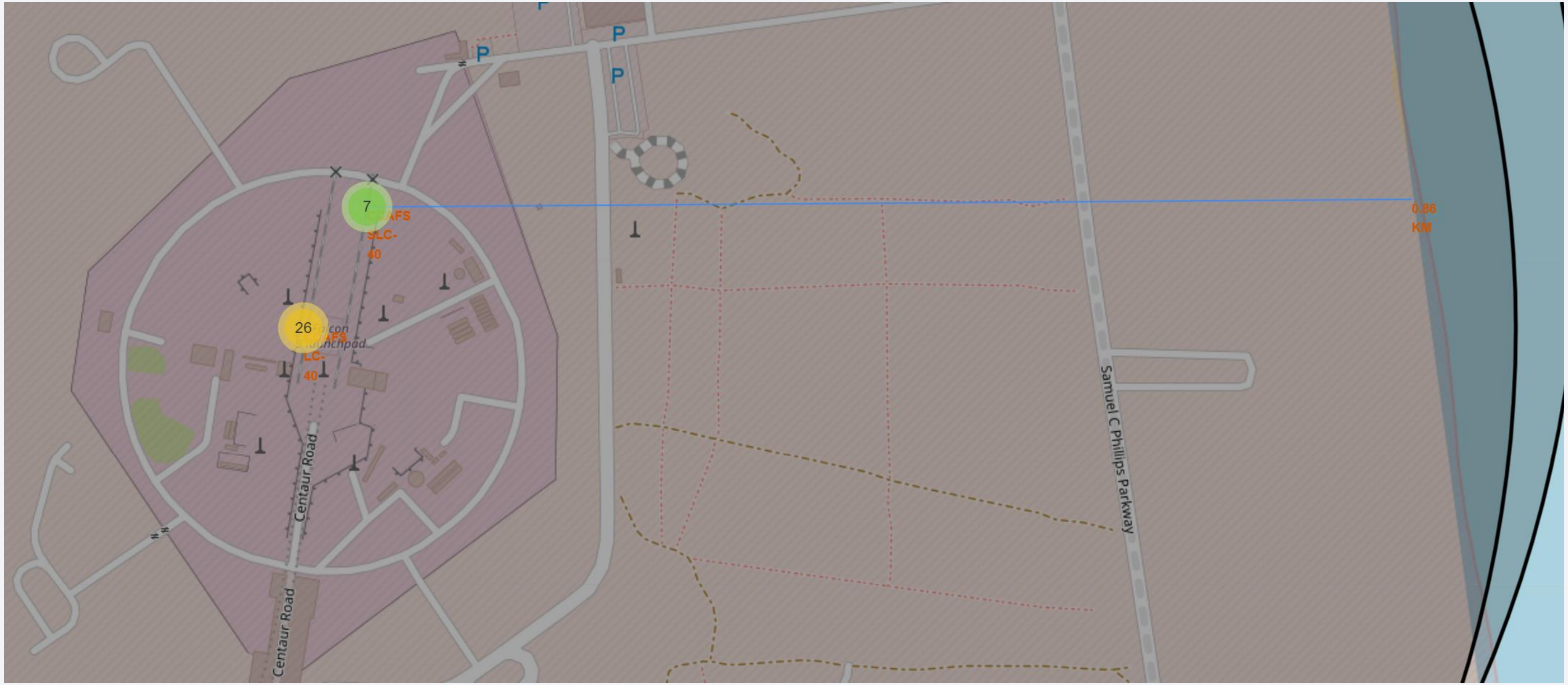
Launch outcomes mapped

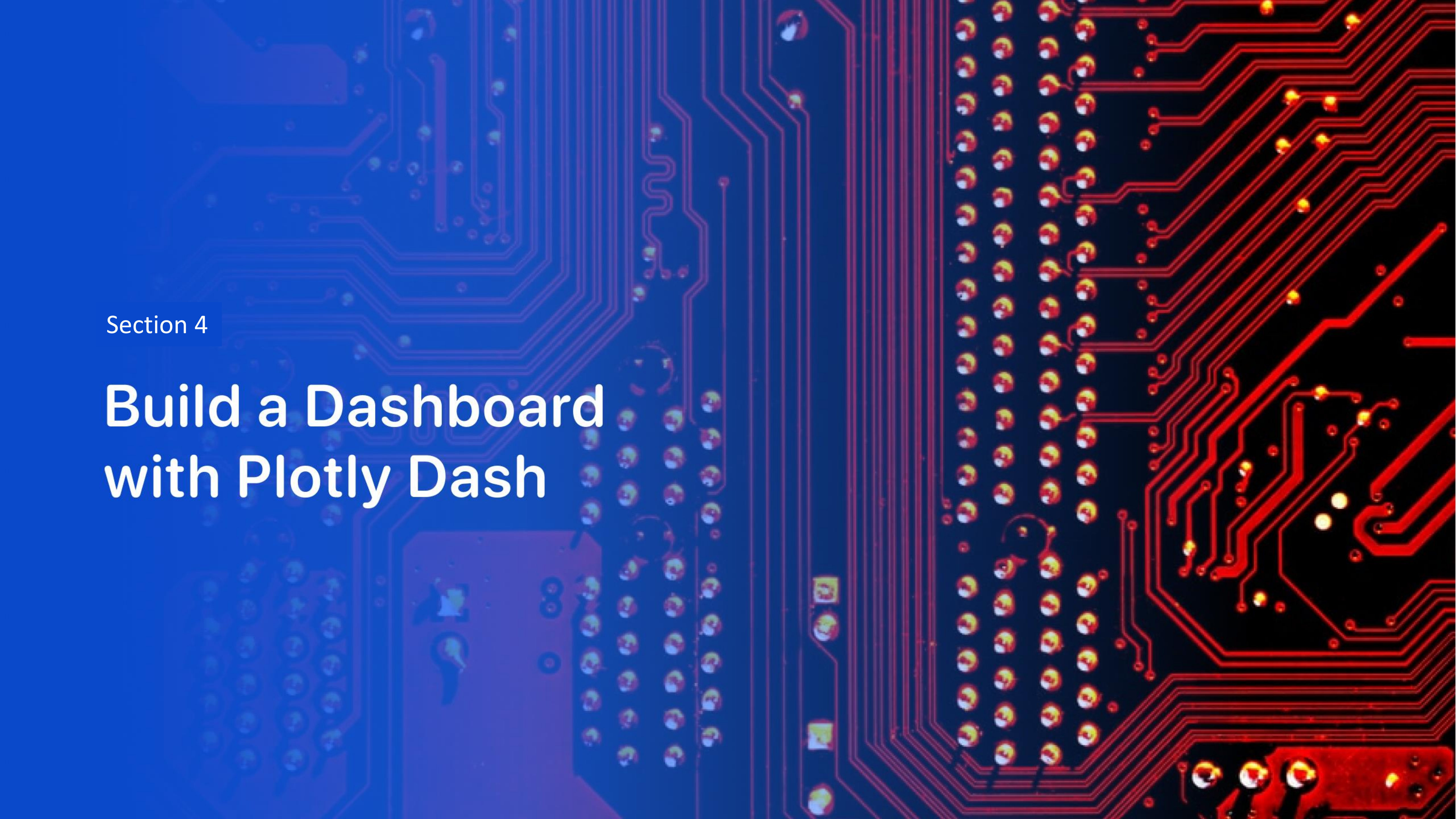
- A launch site cluster expanded, showing all the launches from that site, labeled by the launch outcome. Green rocket marks success, red X marks failure



Launch site distance from coast

- A marker and a line showing the distance from a Launch Site to the coast



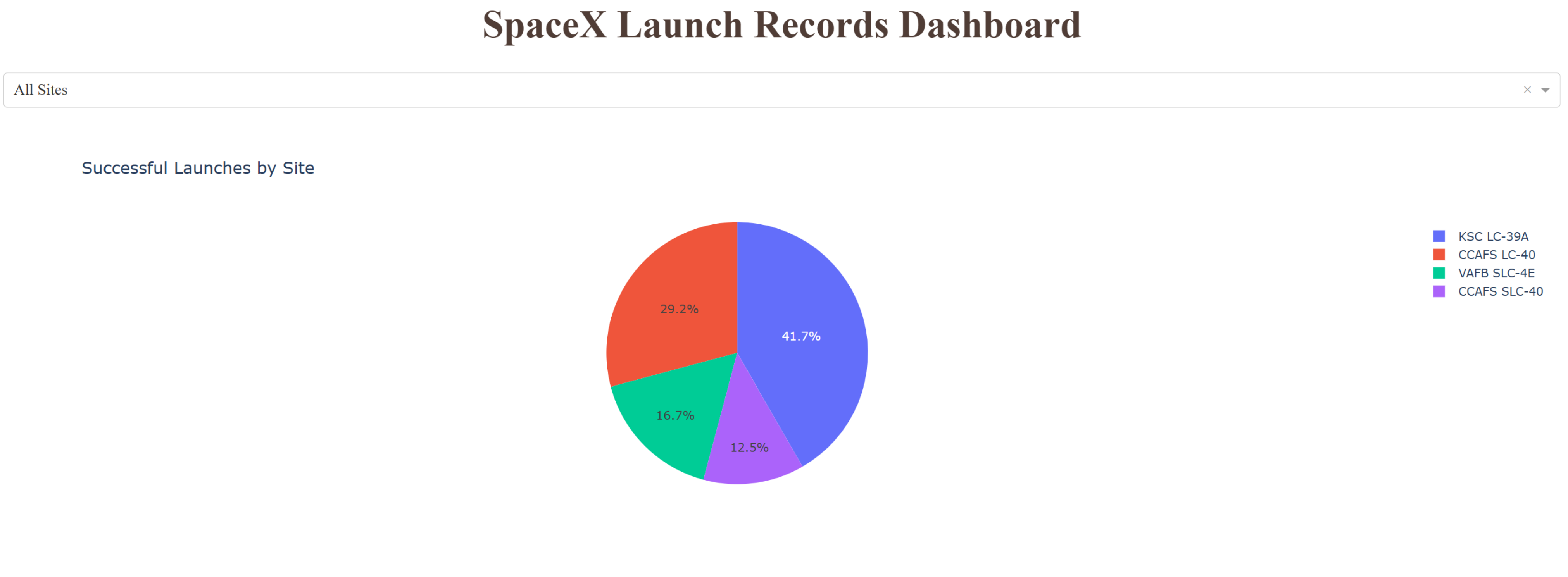


Section 4

Build a Dashboard with Plotly Dash

Launch success pie chart

- A pie chart that shows the distribution of successful launches by launch sites.
- The chart is in an interactive Plotly Dash dashboard

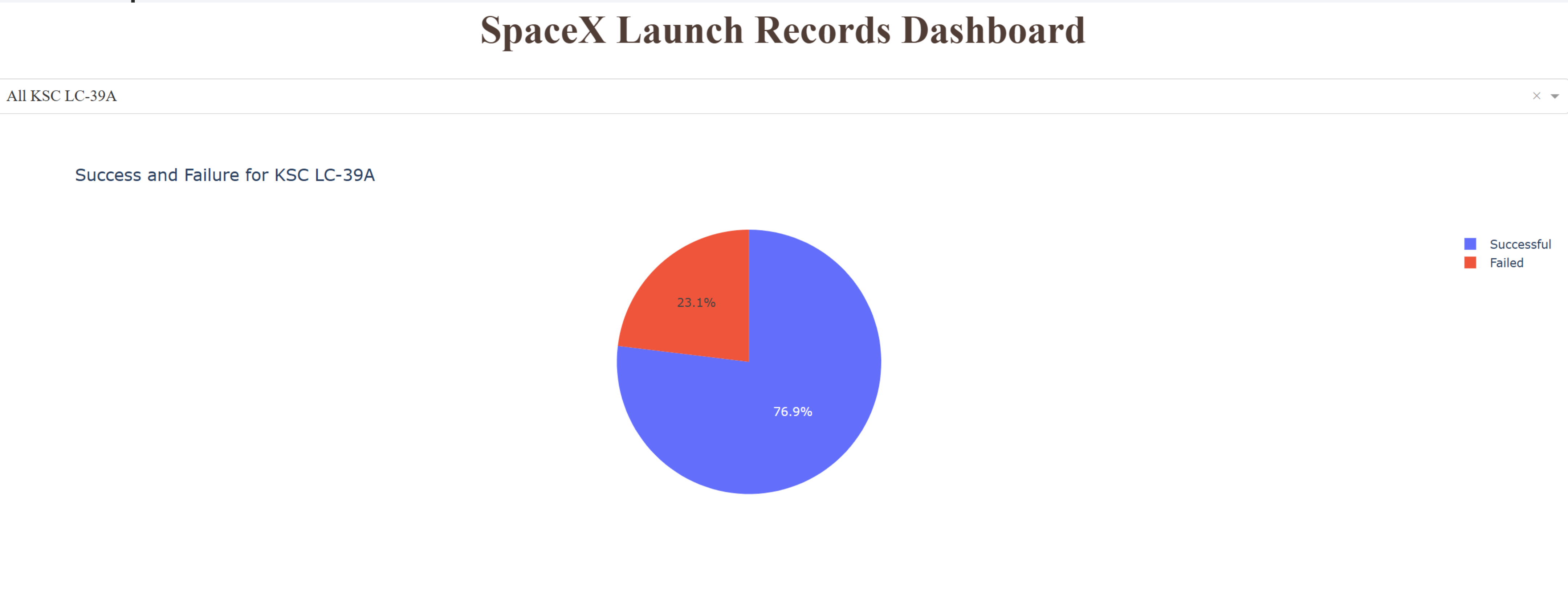


SpaceX Launch Records Dashboard

All Sites

Launch site success ratio

- The same dashboard now focused in on the launch site with the most successful launches
- The pie chart now shows the successful vs failed launches at that launch site

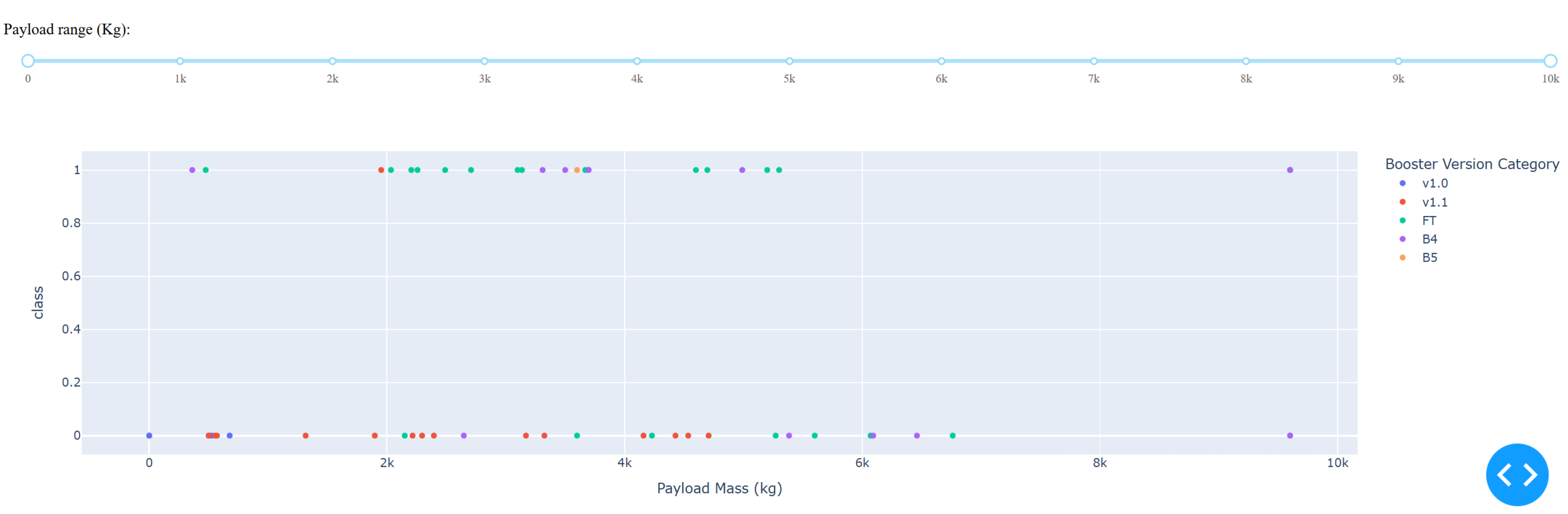


SpaceX Launch Records Dashboard

All KSC LC-39A

Success scatter plot

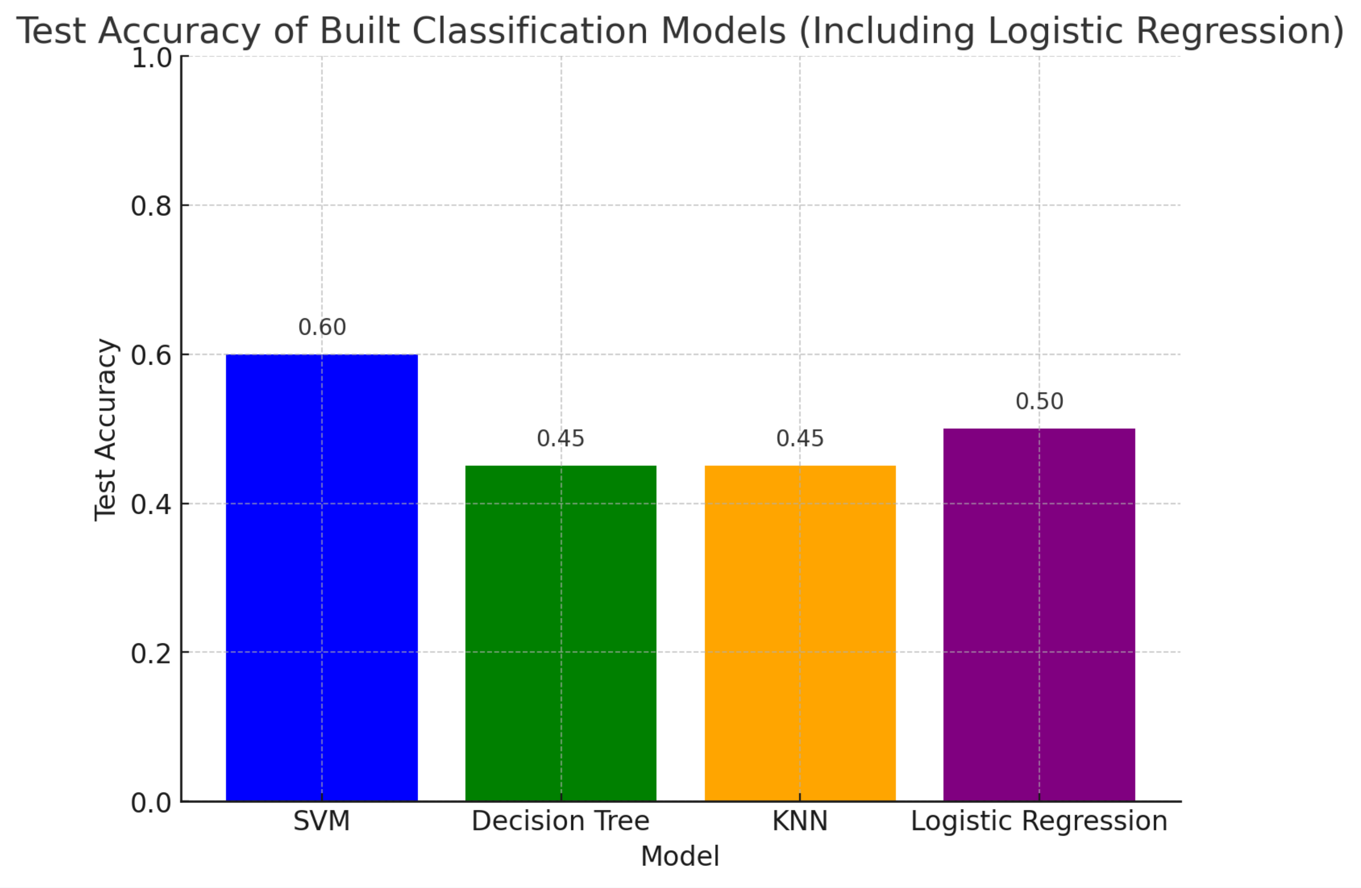
- An interactive scatterplot that has a slider to focus the plot by range of Payload Mass
- This plot also takes effect from the previously shown selection of launch sites.
  - In this screenshot all launch sites have been selected



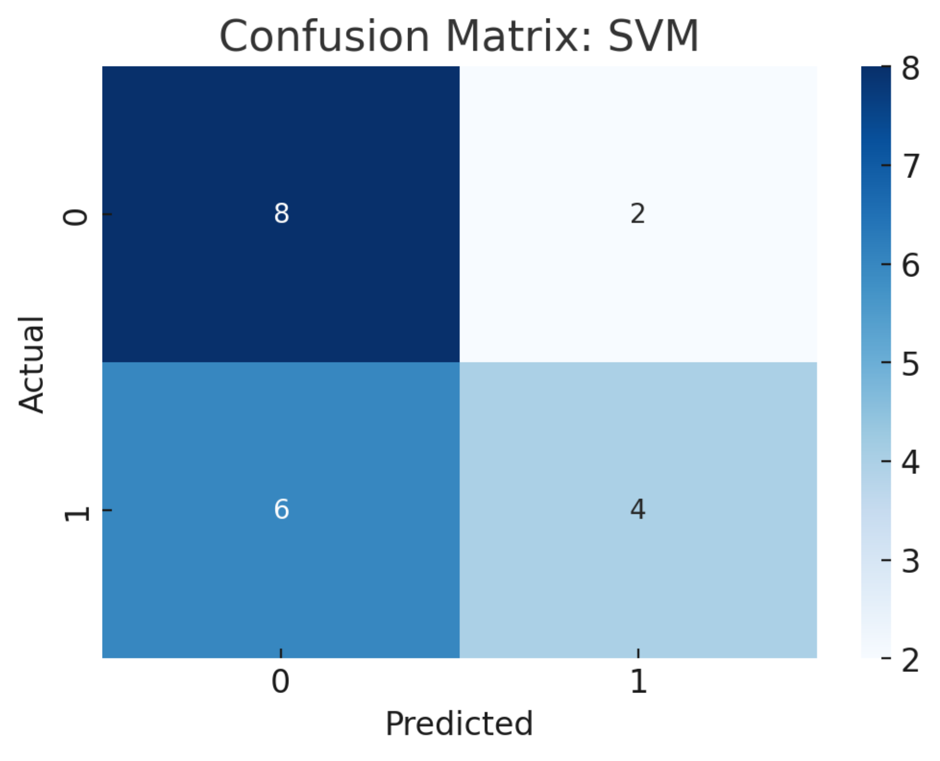
Section 5

Predictive Analysis (Classification)

Classification Accuracy



Confusion Matrix



- The confusion matrix shows that the SVM model correctly predicted 4 positive cases (True Positives) and 8 negative cases (True Negatives).
- The model also made 2 errors by predicting positive for negative cases (False Positives) and 6 errors by predicting negative for positive cases (False Negatives).

Conclusions

- Reusability Impact: The dataset highlights SpaceX's focus on cost savings by enabling reusable rocket stages, reflected in patterns of successful first-stage landings.
- Payload and Launch Success: Heavier payloads tend to have lower success rates, suggesting a tradeoff between payload weight and mission success.
- Geospatial Insights: Launch site proximity to water bodies and specific geographic features significantly influences the choice of launch sites and potentially the success of missions.
- SVM Dominates: The Support Vector Machine (SVM) consistently outperformed other models, achieving the highest test accuracy, indicating its suitability for the binary classification task.

Thank you!