



IBM Developer  
SKILLS NETWORK

# Winning Space Race with Data Science

Ricardo Urdaneta  
July 3, 2025



# Outline

---

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

# Executive Summary

---

## Methodologies

- Data Collection & Wrangling: Data was gathered from the SpaceX API and supplemented with web scraping from Wikipedia. The combined data was then cleaned, and a binary target variable, Class, was engineered to represent successful (1) or failed (0) landings.
- Exploratory Data Analysis (EDA): The dataset was analyzed using SQL queries for specific aggregations and data visualization techniques (scatter, bar, and line plots) to uncover initial trends and relationships between launch variables and outcomes.
- Interactive Analytics: Interactive tools were developed to deepen the analysis. Folium was used to create maps visualizing launch site locations and their proximities, while Plotly Dash was used to build a dynamic dashboard for interactive data filtering and exploration.
- Predictive Analysis: Four machine learning classification models (Logistic Regression, SVM, Decision Tree, and KNN) were built and tuned using GridSearchCV to predict the likelihood of a successful first-stage landing.

## Results

- Key Trends Identified: The EDA revealed a strong, positive correlation between the flight number and the success rate, indicating a clear learning curve over time. Success rates were also found to vary significantly by orbit type and payload mass.
- Model Performance: The predictive analysis was highly successful. After hyperparameter tuning, all four classification models achieved an identical accuracy of 83.33% on the test data, confirming that landing success is highly predictable using the selected features.
- Interactive Insights: The interactive dashboard and maps provided a dynamic way to confirm these trends, allowing for on-the-fly filtering that showed, for example, how success rates improved at specific launch sites or within certain payload mass ranges.

# Introduction

---

SpaceX advertises Falcon 9 rocket launches at a cost of \$62 million, significantly lower than other providers who charge upwards of \$165 million. A major reason for this cost saving is SpaceX's ability to reuse the first stage of the rocket.

By predicting whether the Falcon 9 first stage will land successfully, we can better determine the true cost of a launch.

This predictive information is valuable for any company that wishes to bid against SpaceX for a rocket launch contract, as it allows for a more accurate financial analysis and a more competitive bidding strategy.



Section 1

# Methodology

# Methodology

---

## Executive Summary

- Data collection methodology:
  - Data was gathered from SpaceX API and Web Scraping
- Perform data wrangling
  - Find missing values, calculate numbers of launches on each site, number and occurrences of each orbit, mission outcome of the orbits and create a landing outcome label
- 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 different classification models

# Data Collection

---

- SpaceX API:

Initial data was collected programmatically from the SpaceX v4 API, providing structured information on all historical launches.

- Web Scraping:

To supplement the API data, historical launch records were collected by web scraping the 'List of Falcon 9 and Falcon Heavy launches' page on Wikipedia using BeautifulSoup.

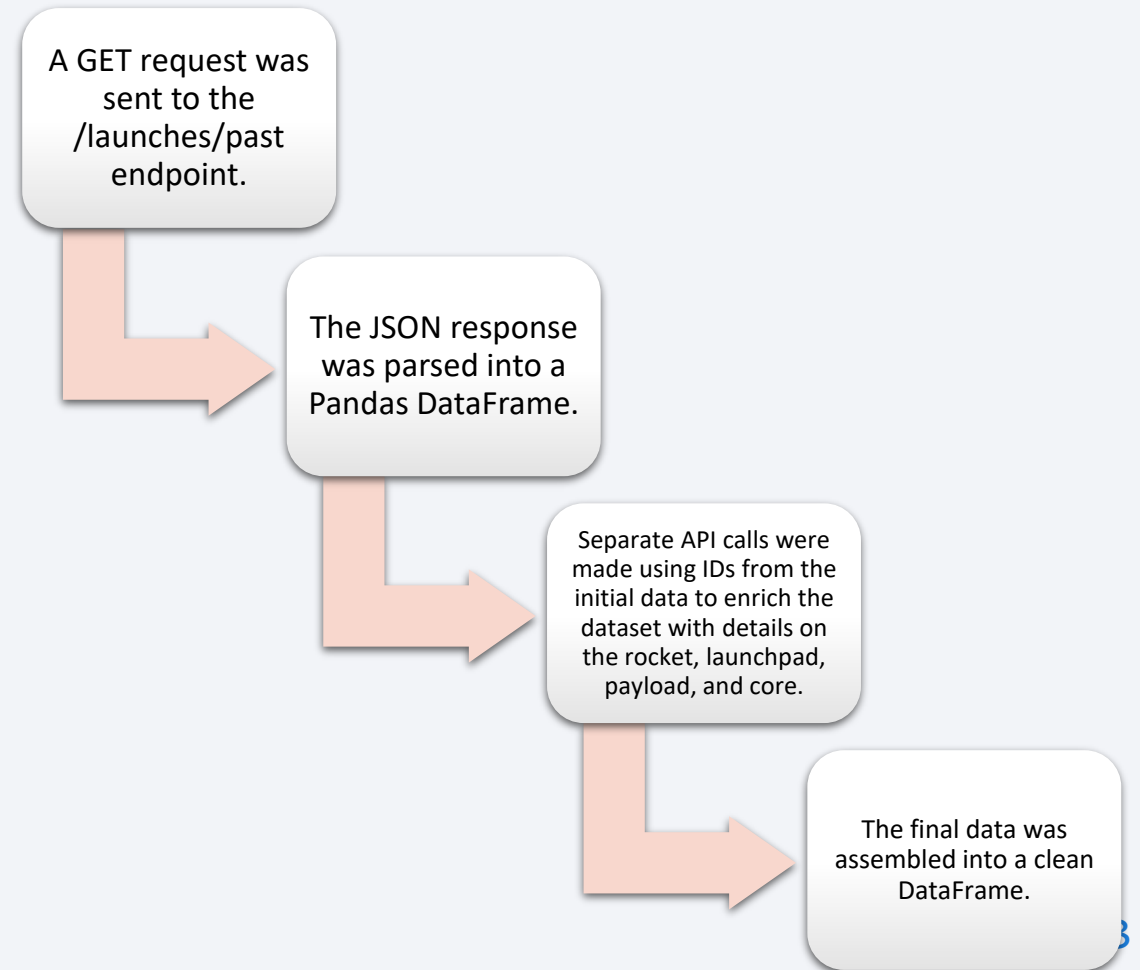
# Data Collection – SpaceX API

---

Initial data was collected programmatically from the SpaceX v4 API, providing structured information on all historical launches.

GitHub URL SpaceX API Notebook:

- <https://github.com/Ricardouchub/Data-Science-Capstone/blob/main/Module%201%20-%20Introduction%20and%20Data%20Wrangling/1%20-%20Lab%20Complete%20the%20Data%20Collection%20API%20Lab.ipynb>
- **GitHub URL Outcome:**
- [https://github.com/Ricardouchub/Data-Science-Capstone/blob/main/Module%201%20-%20Introduction%20and%20Data%20Wrangling/falcon9\\_launches.csv](https://github.com/Ricardouchub/Data-Science-Capstone/blob/main/Module%201%20-%20Introduction%20and%20Data%20Wrangling/falcon9_launches.csv)





# Data Collection - Scraping

---

To supplement the API data, historical launch records were collected by web scraping the 'List of Falcon 9 and Falcon Heavy launches' page on Wikipedia using BeautifulSoup.


## GitHub URL Web Scraping Notebook:

- <https://github.com/Ricardouchub/Data-Science-Capstone/blob/main/Module%201%20-%20Introduction%20and%20Data%20Wrangling/2%20-%20Lab%20Complete%20the%20Data%20Collection%20with%20Web%20Scraping%20lab.ipynb>


## GitHub URL Outcome:

- [https://github.com/Ricardouchub/Data-Science-Capstone/blob/main/Module%201%20-%20Introduction%20and%20Data%20Wrangling/spacex\\_web\\_scraped.csv](https://github.com/Ricardouchub/Data-Science-Capstone/blob/main/Module%201%20-%20Introduction%20and%20Data%20Wrangling/spacex_web_scraped.csv)


The static Wikipedia page's HTML was retrieved using the requests library.



BeautifulSoup was used to parse the HTML content and identify the main launch history table.



The code iterated through each table row to extract



The parsed data was organized into a Pandas DataFrame containing launch data.

# Data Wrangling

---

Data from both sources was cleaned and merged into a single dataset suitable for analysis.

GitHub URL Data Wrangling Notebook:

<https://github.com/Ricardouchub/Data-Science-Capstone/blob/main/Module%201%20-%20Introduction%20and%20Data%20Wrangling/3%20-%20Lab%20Data%20Wrangling%20and%20exploratory%20analysis.ipynb>

The dataset was filtered to include only Falcon 9 launches

Missing PayloadMass values were handled by imputing the column's mean

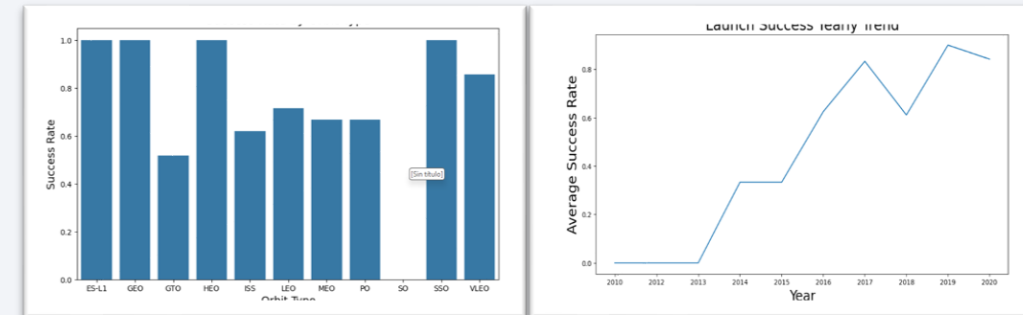
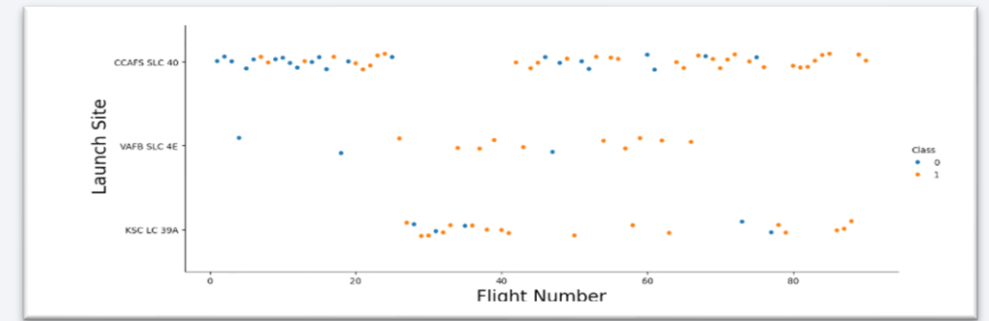
Feature Engineering: The Outcome column, which had various text-based results, was used to create the binary target variable Class. A value of 1 was assigned for successful landings and 0 for unsuccessful ones

The final, labeled dataset was saved for the analysis and modeling phases

# EDA with Data Visualization

To understand the relationships between different launch attributes, a series of visualizations were created using Matplotlib and Seaborn.

- **Scatter Plots:** Used to show the correlation between variables like FlightNumber, PayloadMass, LaunchSite, and Orbit. The landing outcome (Class) was used as a hue to visually distinguish successful from unsuccessful launches.
- **Bar Chart:** Created to compare the success rate across different orbit types.
- **Line Chart:** Used to visualize the trend of the average launch success rate over the years. Add the GitHub URL of your completed EDA with data visualization notebook, as an external reference and peer-review purpose



GitHub URL EDA with Data Visualization Notebook:

[https://github.com/Ricardouchub/Data-Science-Capstone/blob/main/Module%20%20-%20Exploratory%20Data%20Analysis%20\(EDA\)/5%20-%20Lab%20EDA%20with%20Visualization.ipynb](https://github.com/Ricardouchub/Data-Science-Capstone/blob/main/Module%20%20-%20Exploratory%20Data%20Analysis%20(EDA)/5%20-%20Lab%20EDA%20with%20Visualization.ipynb)

# EDA with SQL

---

SQL queries were used to directly query and aggregate the data to answer specific analytical questions.

GitHub URL EDA with SQL Notebook:

[https://github.com/Ricardouchub/Data-Science-Capstone/blob/main/Module%20%20-%20Exploratory%20Data%20Analysis%20\(EDA\)/4%20-%20%20Lab%20Complete%20the%20EDA%20with%20SQL.ipynb](https://github.com/Ricardouchub/Data-Science-Capstone/blob/main/Module%20%20-%20Exploratory%20Data%20Analysis%20(EDA)/4%20-%20%20Lab%20Complete%20the%20EDA%20with%20SQL.ipynb)

Retrieved unique launch sites

Filtered for launches from 'CCA' sites

Calculated total and average payload mass for specific customers and booster versions

Identified key mission milestones, such as the first successful ground pad landing and boosters that carried the maximum payload

Counted total mission successes and failures

Analyzed failed landings for a specific year (2015)

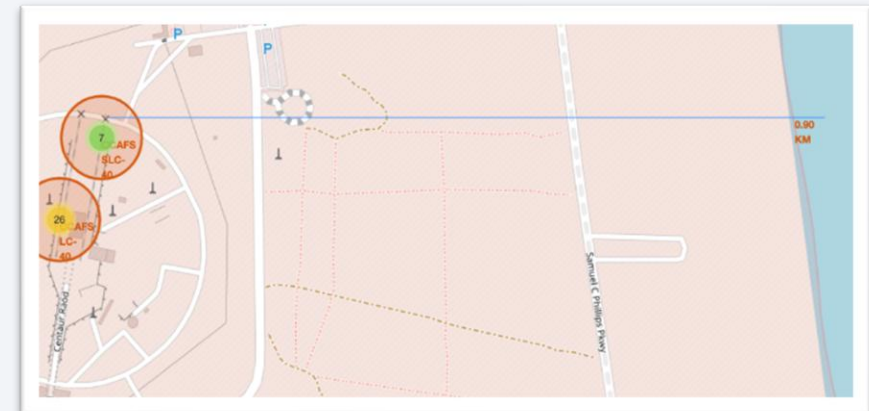
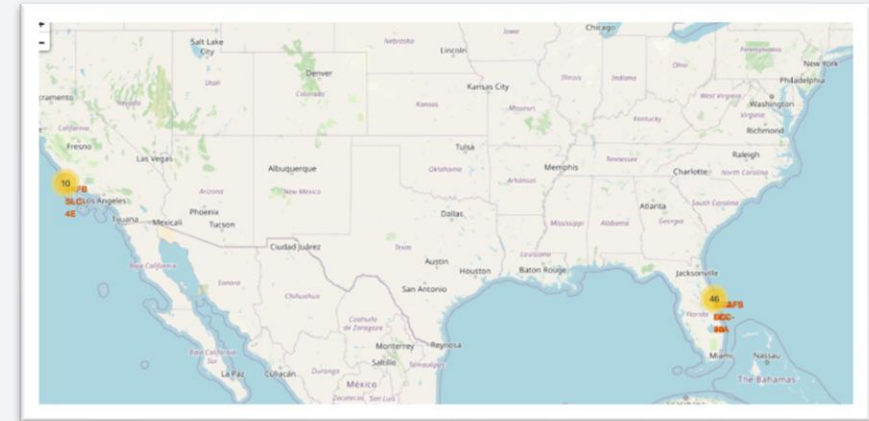
# Build an Interactive Map with Folium

An interactive map was built using Folium to visualize the geographic aspects of the launch data.

- **Launch Site Markers:** Markers were added for each launch site with popups displaying the site's name.
- **Proximity Lines & Markers:** For a selected launch site, markers were added for nearby railways, highways, and coastlines. Lines were drawn from the launch pad to these points with the calculated distance displayed in the marker's popup.

GitHub URL Folium Notebook:

[https://github.com/Ricardouchub/Data-Science-Capstone/blob/main/Module%203%20-%20Interactive%20Visual%20Analytics%20with%20Folium%20lab%20andd%20Dashboard%20with%20Plotly%20Dash/lab\\_jupyter\\_launch\\_site\\_location.jupyterlite.ipynb](https://github.com/Ricardouchub/Data-Science-Capstone/blob/main/Module%203%20-%20Interactive%20Visual%20Analytics%20with%20Folium%20lab%20andd%20Dashboard%20with%20Plotly%20Dash/lab_jupyter_launch_site_location.jupyterlite.ipynb)

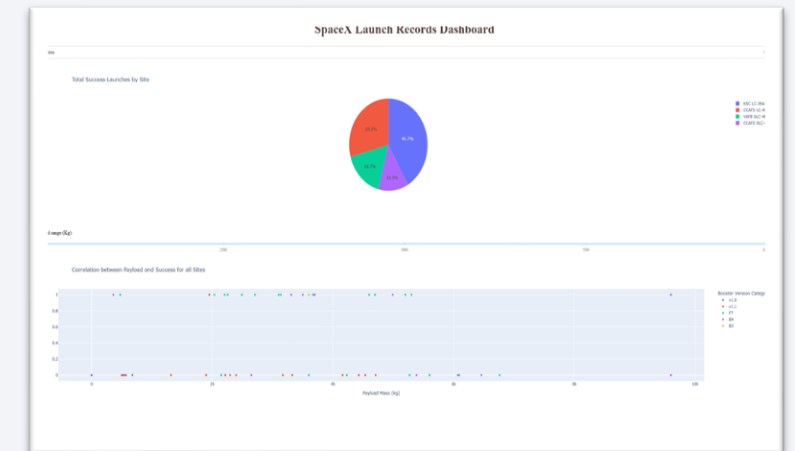




# Build a Dashboard with Plotly Dash

A web-based interactive dashboard was created using Plotly Dash to allow for dynamic data exploration with the following components:

- **Dropdown Menu:** Allows users to filter all charts by a specific launch site or view all sites.
- **Pie Chart:** Visualizes the total count of successful versus failed launches for the selected site(s).
- **Scatter Plot:** Correlates PayloadMass with the launch Class (outcome).
- **Payload Mass Range Slider:** Filters the scatter plot based on a selected payload mass range.



GitHub URL Plotly Dash App:

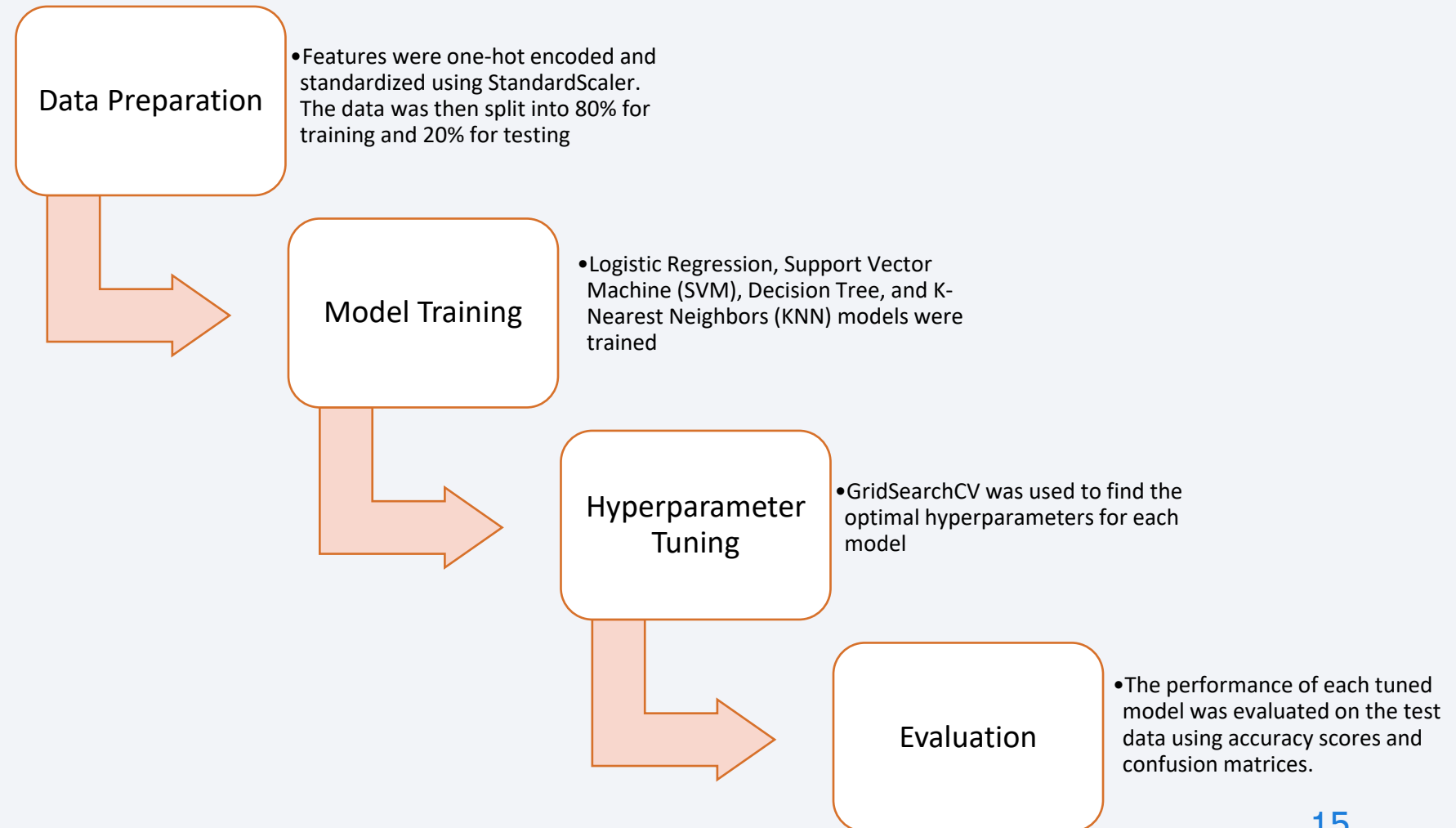
<https://github.com/Ricardouchub/Data-Science-Capstone/blob/main/Module%203%20-%20Interactive%20Visual%20Analytics%20with%20Folium%20lab%20andd%20Dashboard%20with%20Plotly%20Dash/spacex-dash-app.py>

# Predictive Analysis (Classification)

The final phase was to build classification models to predict landing success using four different algorithms.

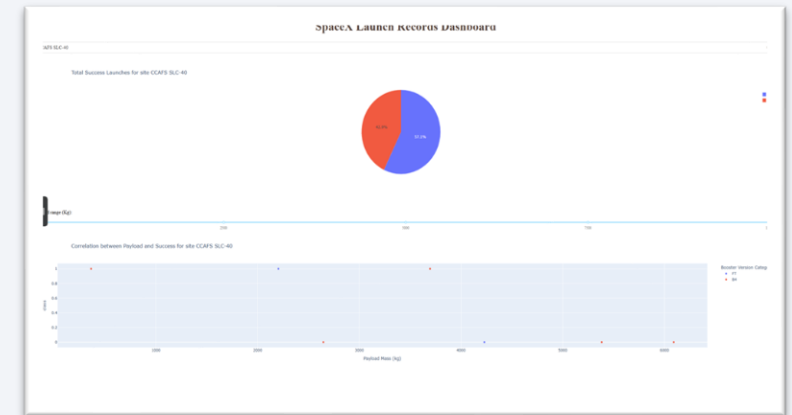
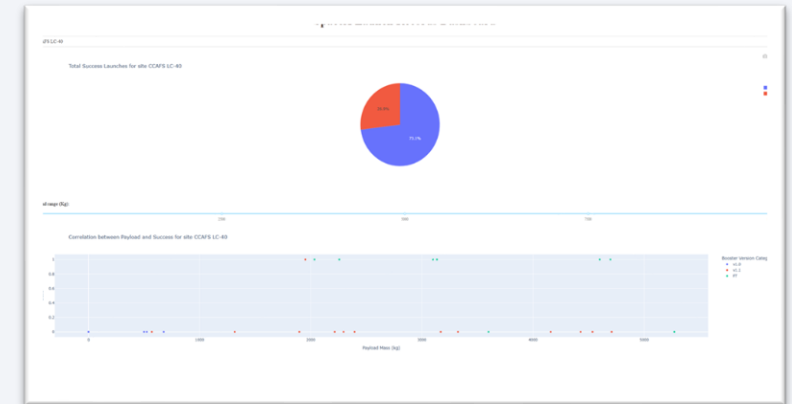
GitHub URL Predictive Analysis Notebook:

[https://github.com/Ricardouchub/Data-Science-Capstone/blob/main/Module%204%20-%20Predictive%20Analysis%20\(Classification\)/SpaceX%20Machine%20Learning%20Prediction.ipynb](https://github.com/Ricardouchub/Data-Science-Capstone/blob/main/Module%204%20-%20Predictive%20Analysis%20(Classification)/SpaceX%20Machine%20Learning%20Prediction.ipynb)



# Results

- *Exploratory data analysis* revealed that launch success is heavily influenced by several key factors. A strong positive correlation was found between the flight number and the success rate, demonstrating a clear learning curve over time. Additionally, success rates vary significantly by orbit type, with some orbits proving more challenging than others, while SQL queries helped quantify these findings and pinpoint specific mission milestones.
- *The predictive analysis* confirmed that landing outcomes are highly predictable. After training and tuning four different classification models, a consistent and high accuracy of 83.33% was achieved on the test data. This result validates that the features selected during the analysis—such as booster version, orbit, and flight history—are strong predictors of whether a Falcon 9 first stage will land successfully.





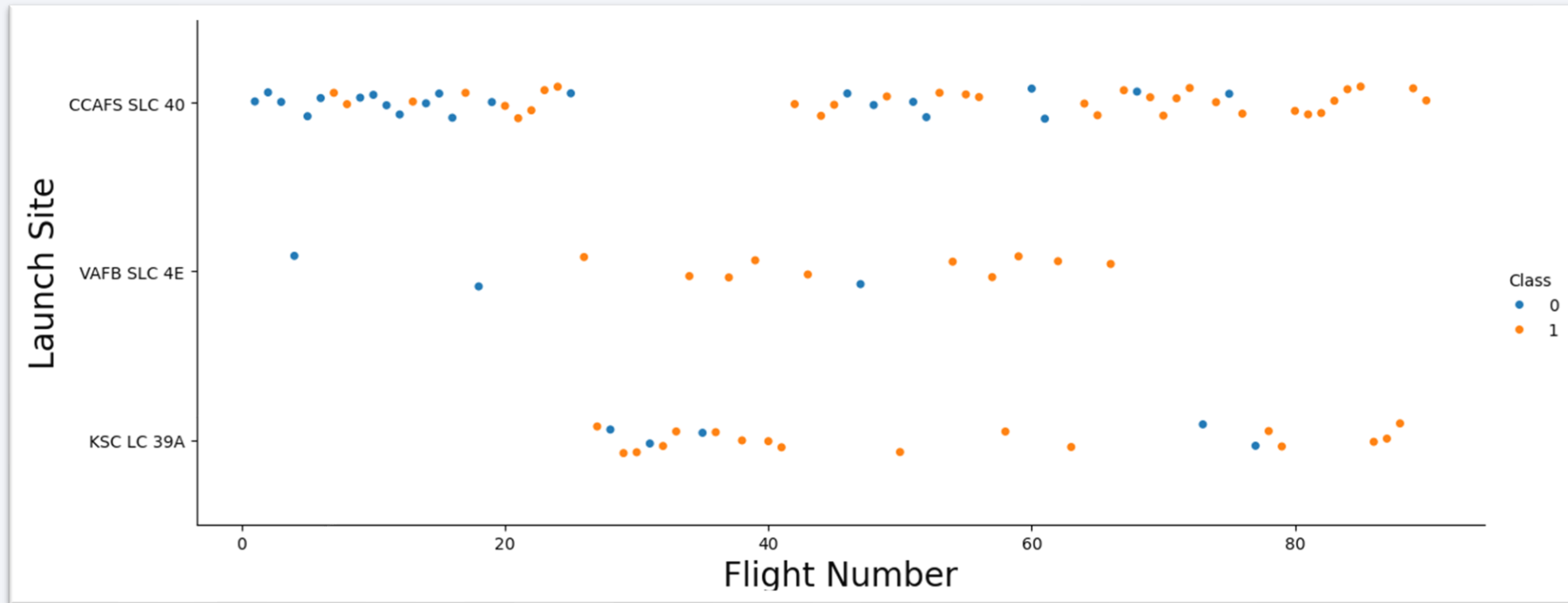
The background of the slide is an abstract composition. It features a dark blue base color. Overlaid on this are numerous diagonal streaks in shades of red and cyan. A faint, light blue grid pattern is also visible, particularly in the lower half of the image. The overall effect is dynamic and technological.

Section 2

# Insights drawn from EDA



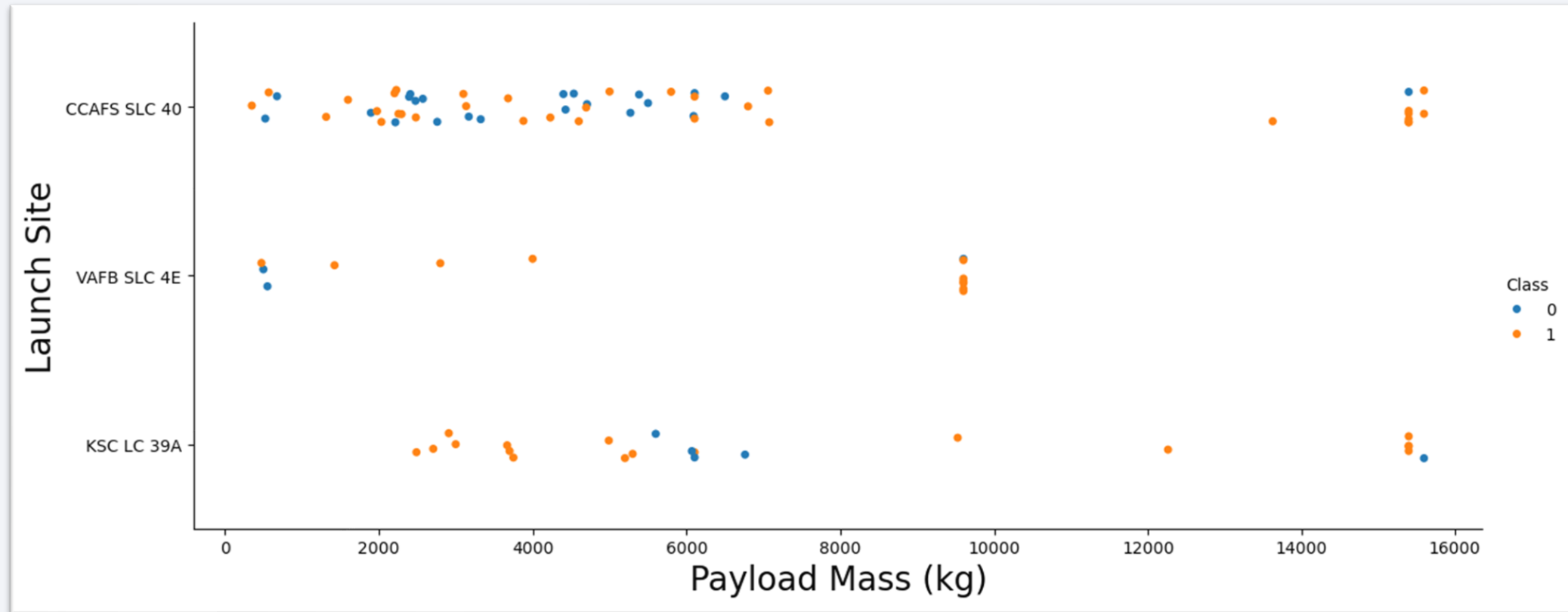
# Flight Number vs. Launch Site



The plot clearly shows that as the flight number increases, the success rate improves across all launch sites. Early launches show a mixture of successes and failures, while later launches are almost all successful. This indicates a strong learning curve and increasing reliability.



# Payload vs. Launch Site

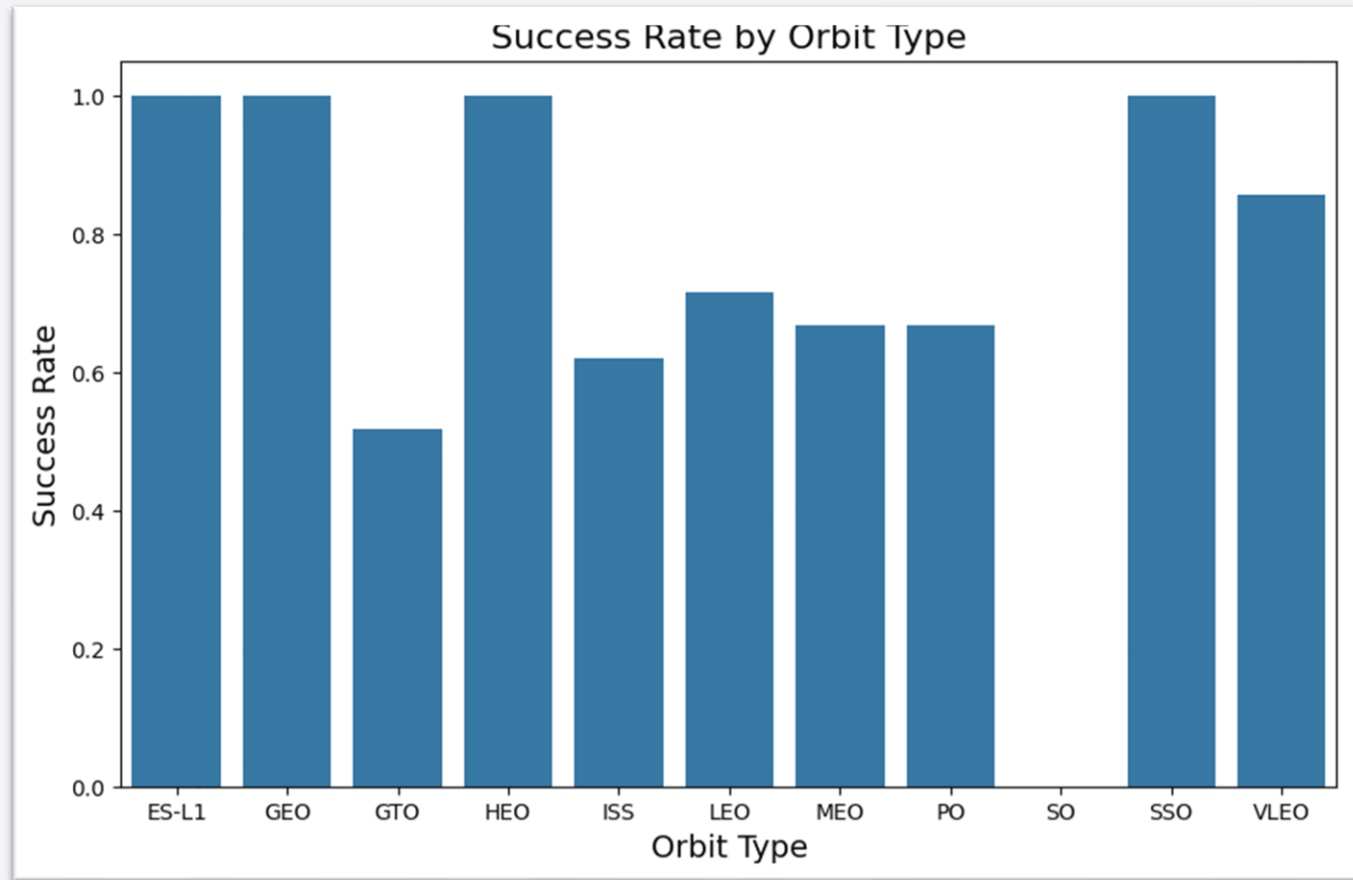


The VAFB SLC 4E site is used for launches with lower payload masses. In contrast, the CCAFS SLC 40 and KSC LC 39A sites handle a much wider range of payloads, including the heaviest ones (over 10,000 kg).

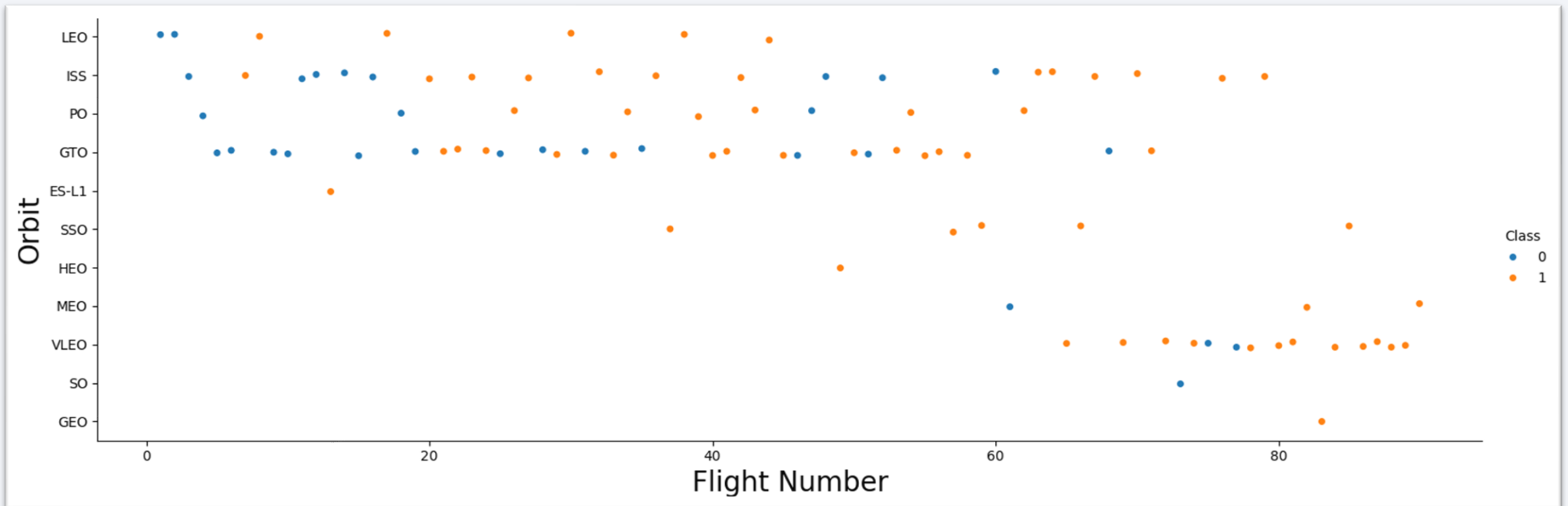
# Success Rate vs. Orbit Type

---

The chart reveals that orbits like ES-L1, GEO, HEO, and SSO have achieved a 100% success rate. The GTO orbit, a common and challenging destination, has a noticeably lower success rate in comparison.

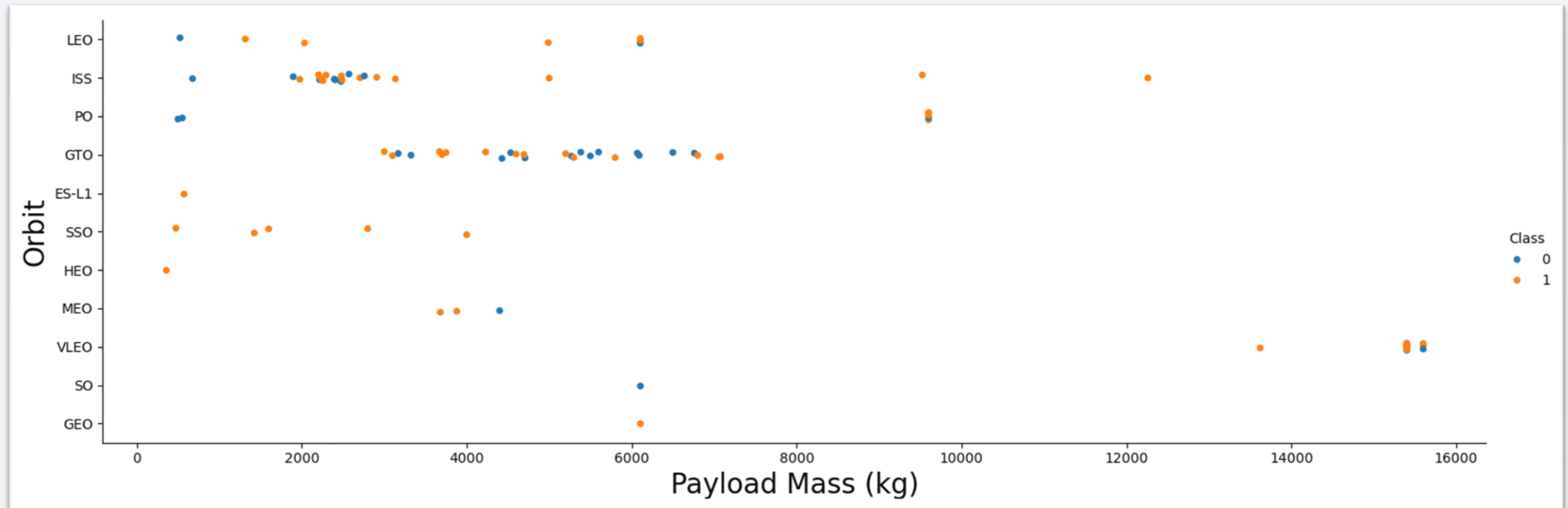


# Flight Number vs. Orbit Type



This plot confirms that success in most orbits is correlated with flight number. For orbits like GTO and ISS, which have more launches, the progression from early failures to later successes is clearly visible.

# Payload vs. Orbit Type

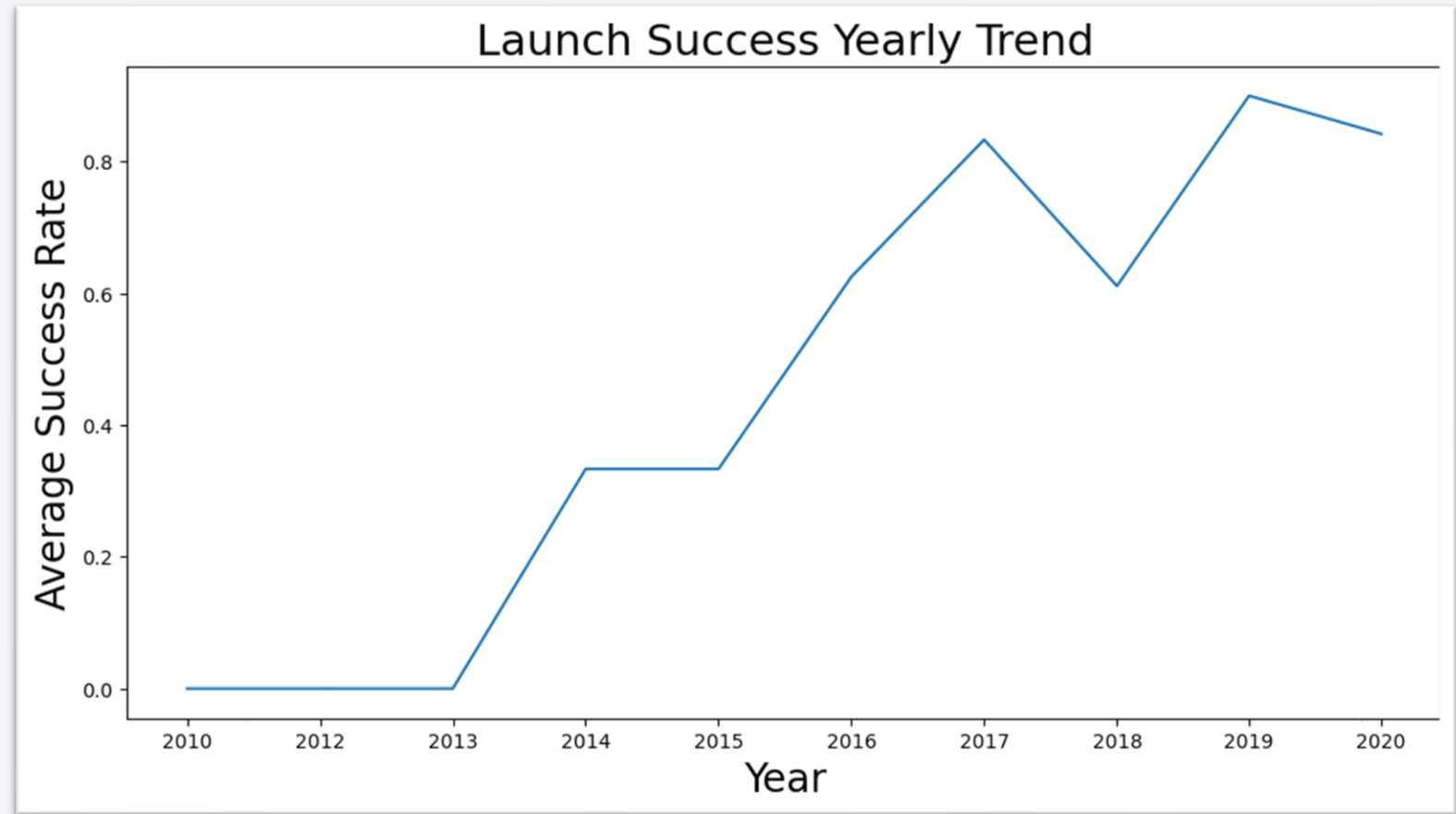


This visualization highlights that successful missions to high-demand orbits like VLEO, ISS, and GTO have been achieved across a wide spectrum of payload masses. For VLEO missions with very heavy payloads ~15,600 kg, the success rate is remarkably high.

# Launch Success Yearly Trend

---

The chart illustrates a clear positive trend in the average success rate from 2010 to 2020. After initial challenges, the success rate has steadily climbed, demonstrating significant improvements in technology and reliability.





# All Launch Site Names

---

```
%sql SELECT DISTINCT "Launch_Site" FROM SPACEXTABLE;
```

This query uses the SELECT DISTINCT statement to retrieve and display all the unique names from the "Launch\_Site" column in the SPACEXTABLE. It ensures that each launch site appears only once in the result, even if it's listed multiple times in the table.

Query Result:

CCAFS LC-40

VAFB SLC-4E

KSC LC-39A

CCAFS SLC-40

```
%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'

## Task 2

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

```
%sql SELECT * FROM SPACEXTABLE 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

```
%sql SELECT * FROM SPACEXTABLE WHERE "Launch_Site" LIKE 'CCA%' LIMIT 5;
```

This query is designed to filter the dataset for specific records based on a pattern. The LIKE operator is used in the WHERE clause to search for a specified pattern. 'CCA%' will find any values in the "Launch\_Site" column that start with "CCA". The % is a wildcard that matches any following characters. The LIMIT 5 clause then restricts the output to the first 5 records found.

# Total Payload Mass

---

```
%sql SELECT SUM("PAYLOAD_MASS__KG_")  
FROM SPACEXTABLE WHERE "Customer" =  
'NASA (CRS)';
```

This query uses the SUM() aggregate function to calculate the total value of a numeric column. The WHERE clause filters the data to include only the rows where the "Customer" is 'NASA (CRS)'. As a result, the query adds up the payload mass for all NASA (CRS) missions and returns a single value representing the total mass.

Query result:

45596

```
%sql SELECT SUM("PAYLOAD_MASS__KG_") FROM SPACEXTABLE WHERE "Customer" = 'NASA (CRS)';  
[9] ✓ 0.0s  
... * sqlite:///my\_data1.db  
... Done.  
...  
SUM(PAYLOAD_MASS__KG_)  
45596
```

# Average Payload Mass by F9 v1.1

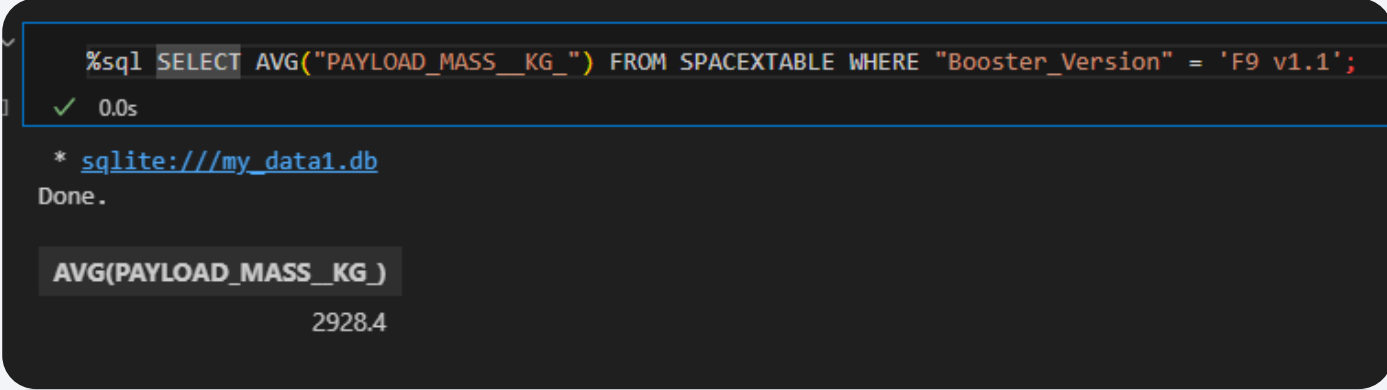
---

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

This query uses the `AVG()` function to find the average of the `"PAYLOAD_MASS__KG_"` column. The `WHERE` clause filters the data to only include records where the `"Booster_Version"` is 'F9 v1.1', so the average is calculated exclusively for that specific booster type.

Query result:

2928.4



```
%sql SELECT AVG("PAYLOAD_MASS__KG_") FROM SPACEXTABLE WHERE "Booster_Version" = 'F9 v1.1';  
✓ 0.0s  
* sqlite:///my_data1.db  
Done.  
  
AVG(PAYLOAD_MASS__KG_)  
2928.4
```

# First Successful Ground Landing Date

---

```
%sql SELECT MIN("Date") FROM  
SPACEXTABLE WHERE  
"Landing_Outcome" = 'Success  
(ground pad)';
```

This query uses the MIN() aggregate function to find the earliest date in the "Date" column. The WHERE clause restricts this search to only those rows where the "Landing\_Outcome" was a 'Success (ground pad)'.

Query result:

2015-12-22

```
%sql SELECT MIN("Date") FROM SPACEXTABLE WHERE "Landing_Outcome" = 'Success (ground pad)';  
✓ 0.0s  
* sqlite:///my\_data1.db  
Done.  
  
MIN(Date)  
2015-12-22
```



# Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql SELECT "Booster_Version" FROM  
SPACEXTABLE WHERE "Landing_Outcome" =  
'Success (drone ship)' AND  
"PAYLOAD_MASS_KG_" BETWEEN 4001 AND  
5999;
```

This query retrieves the names of booster versions that meet two specific conditions simultaneously. The WHERE clause is used with an AND operator to combine the filters. The first condition, "Landing\_Outcome" = 'Success (drone ship)', selects only successful drone ship landings. The second condition, "PAYLOAD\_MASS\_KG\_" BETWEEN 4001 AND 5999, further filters those results to include only missions that carried that payload mass.

Query result:

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

```
%sql SELECT "Booster_Version" FROM SPACEXTABLE WHERE "Landing_Outcome" = 'Success (drone ship)' AND "PAYLOAD_MASS_KG_" BETWEEN 4001 AND 5999;
```

```
* sqlite:///my\_data1.db  
Done.
```

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

F9 FT B1031.2

F9 FT B1031.2

# Total Number of Successful and Failure Mission Outcomes

```
%sql SELECT  
"Mission_Outcome", COUNT(*) as  
Total FROM SPACEXTABLE  
GROUP BY "Mission_Outcome";
```

This query uses the GROUP BY clause to categorize all the records based on their "Mission\_Outcome". The COUNT(\*) function then counts the number of entries within each of these distinct groups. This provides a summary of how many launches resulted in success, failure, or other outcomes.

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

```
%sql SELECT "Mission_Outcome", COUNT(*) as Total FROM SPACEXTABLE GROUP BY "Mission_Outcome";  
✓ 0.0s  
* sqlite:///my\_data1.db  
Done.
```

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

# Boosters Carried Maximum Payload

```
%sql SELECT "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

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

This query uses a subquery to first find the single highest value in the "PAYLOAD\_MASS\_\_KG\_" column using MAX(). Then retrieves the "Booster\_Version" for all records where the payload mass equals that maximum value, effectively listing all boosters that have carried the heaviest payload.

Query Result: 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

```
18] %sql SELECT substr("Date", 6, 2) as Month, "Landing_Outcome", "Booster_Version", "Launch_Site" FROM SPACEXTABLE WHERE substr("Date", 0, 5) = '2015' AND "Landing_Outcome" = 'Failure (drone ship)';
Python

* 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
```

```
%sql SELECT substr("Date", 6, 2) as Month, "Landing_Outcome", "Booster_Version", "Launch_Site" FROM SPACEXTABLE
WHERE substr("Date", 0, 5) = '2015' AND "Landing_Outcome" = 'Failure (drone ship)';
```

This query filters the launch records to find specific mission failures within a given year. The WHERE clause uses two conditions: first, it uses the `substr()` function to isolate launches that occurred in the year '2015'; second, it filters those results to only include records where the "Landing\_Outcome" was a 'Failure (drone ship)'. The query then selects and displays the month, outcome, booster version, and launch site for these specific missions.

Query result:

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 WHERE "Date" BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY "Landing_Outcome" ORDER BY Count DESC;
```

\* [sqlite:///my\\_data1.db](#)

Done.

Landing_Outcome	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

Precluded (drone ship) 1  
Failure (parachute) 2  
Uncontrolled (ocean) 3

```
%sql SELECT "Landing_Outcome", COUNT(*) as Count FROM SPACEXTABLE WHERE "Date" BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY "Landing_Outcome" ORDER BY Count DESC;
```

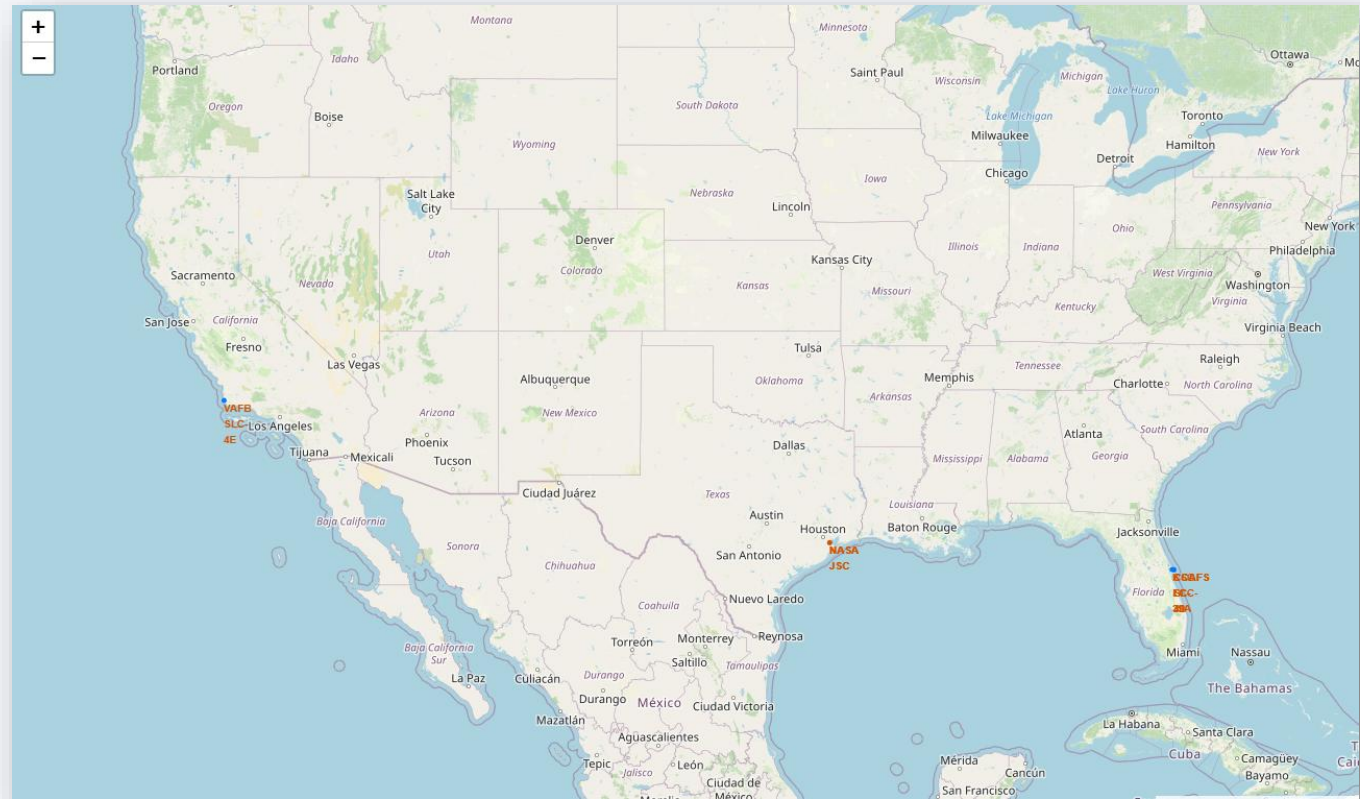
This query first filters the dataset to include only launches between June 4, 2010, and March 20, 2017. It then groups the records by their "Landing\_Outcome", counts the occurrences of each outcome, and finally orders the results from the most frequent to the least frequent outcome.

A satellite view of Earth from space, showing the curvature of the planet and city lights at night. The image is a composite of a solid blue rectangle on the left and a satellite photograph of Earth on the right. The Earth's surface is dark, with numerous bright yellow and orange lights representing cities and urban areas. The horizon of the Earth is visible, separating the dark surface from the deep blue of the atmosphere and the blackness of space.

Section 3

# Launch Sites Proximities Analysis

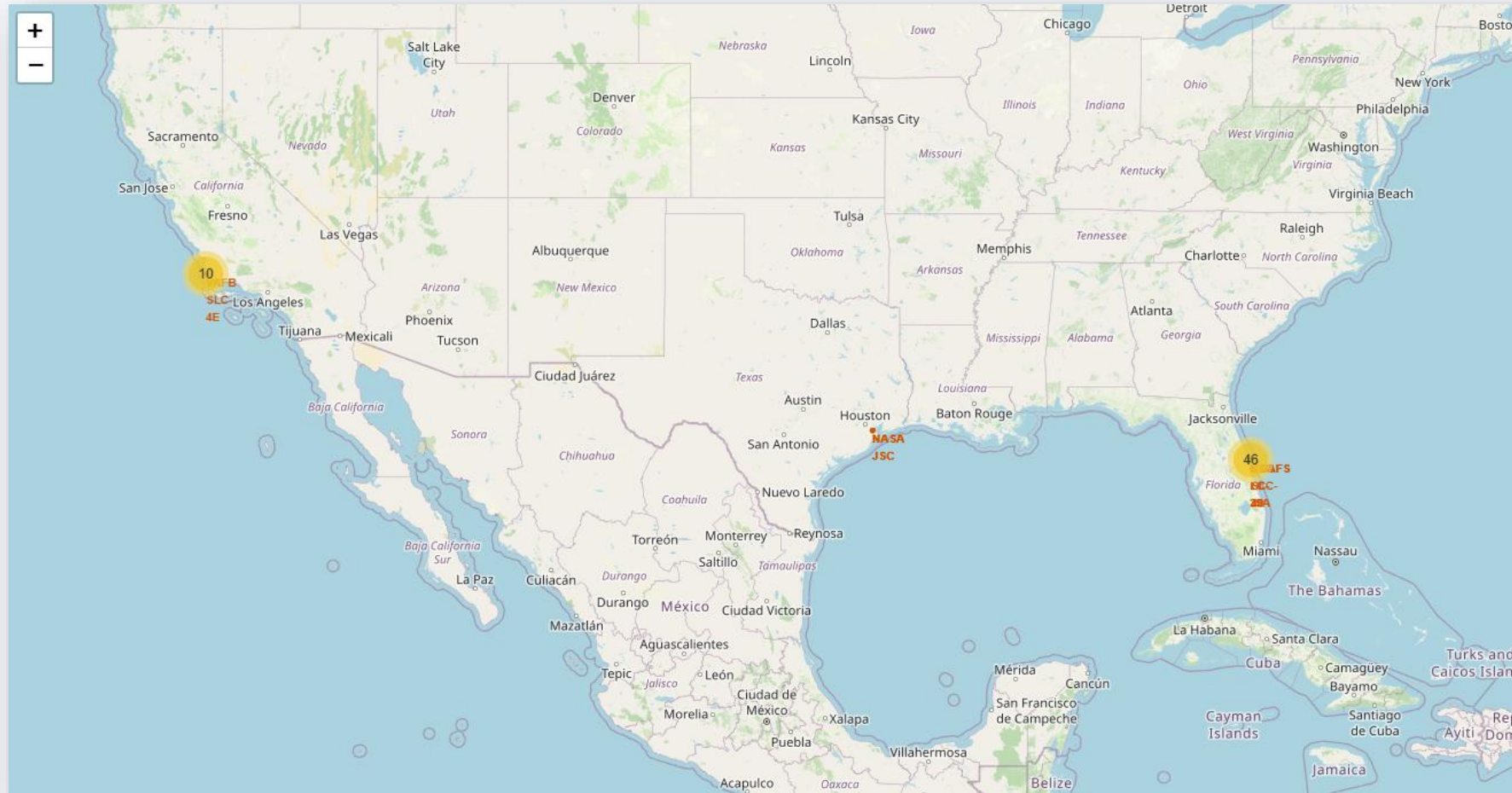
# Global Distribution of SpaceX Launch Sites



This map displays the locations of all four SpaceX launch sites. The geographic clustering on the coasts is strategic for launching rockets over the ocean and away from populated areas.

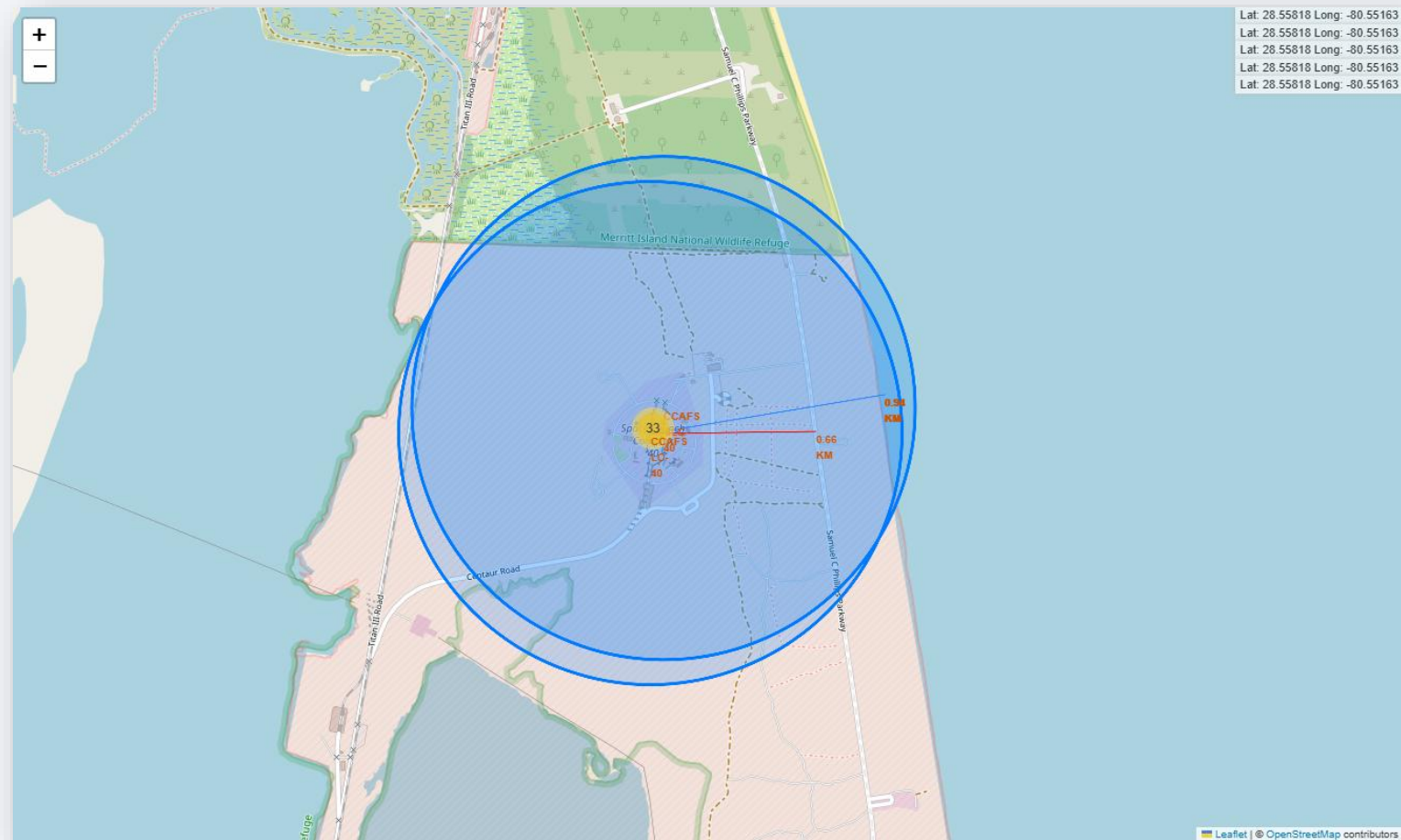


# Launch Success/Failure Visualization at CCAFS SLC-40



This view focuses on a single launch site and uses color-coded markers (green for success, red for failure) to differentiate launch outcomes. The visual density of green markers reinforces the trend of increasing launch reliability.

# Proximity Analysis of VAFB SLC-4E



This map demonstrates the analytical capability of combining location data with geographical features. The lines show the calculated distances from the launch pad to the nearest coastline, highway, and railway. Key parameters for safety and logistics.

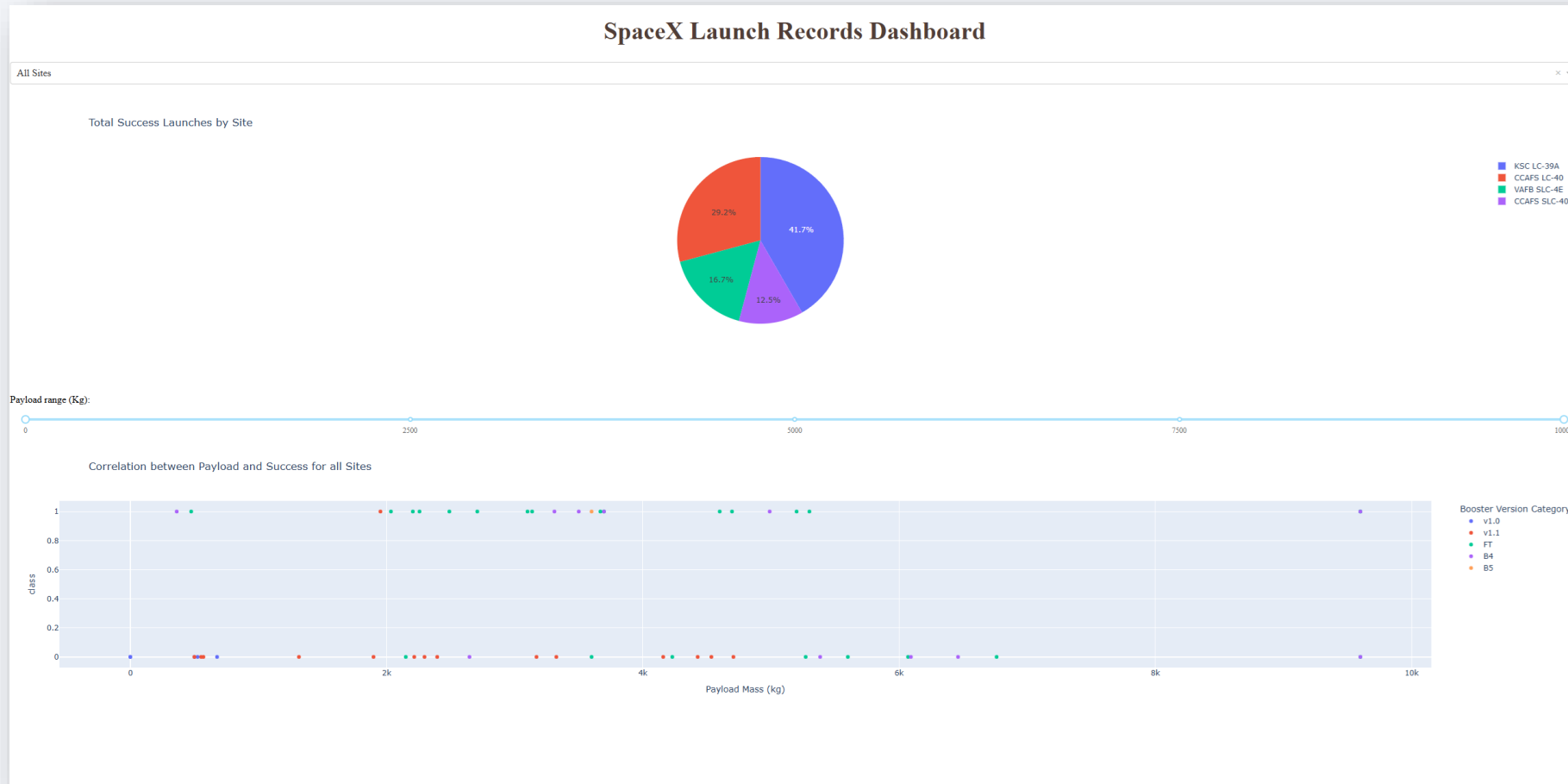




Section 4

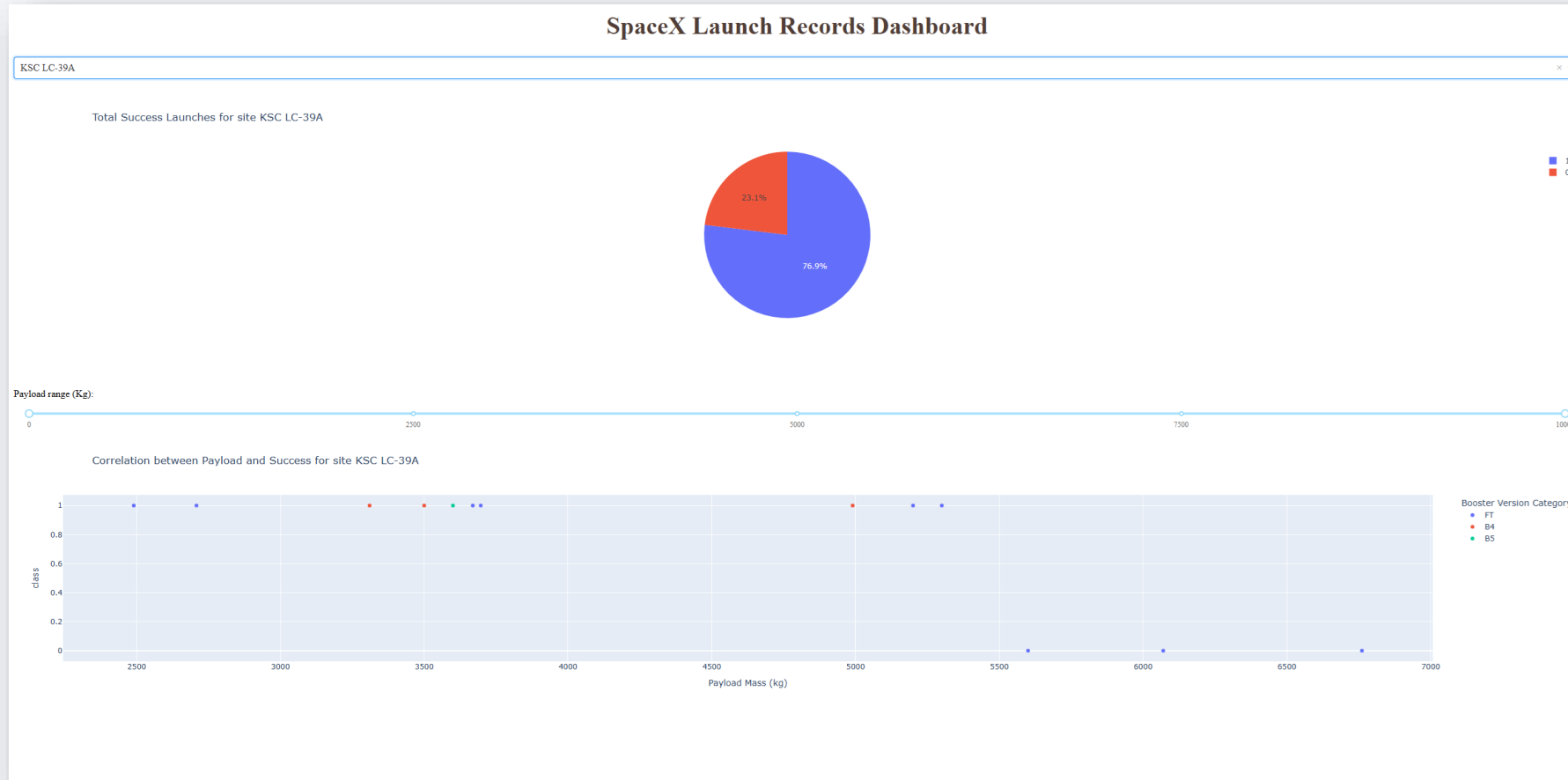
# Build a Dashboard with Plotly Dash

# Overall Launch Success Rate (All Sites)



When "All Sites" selected, this pie chart provides a high-level overview, showing the total count of successful launches. The large green slice immediately communicates the program's high overall success rate.

# Dashboard: Success Rate by Site

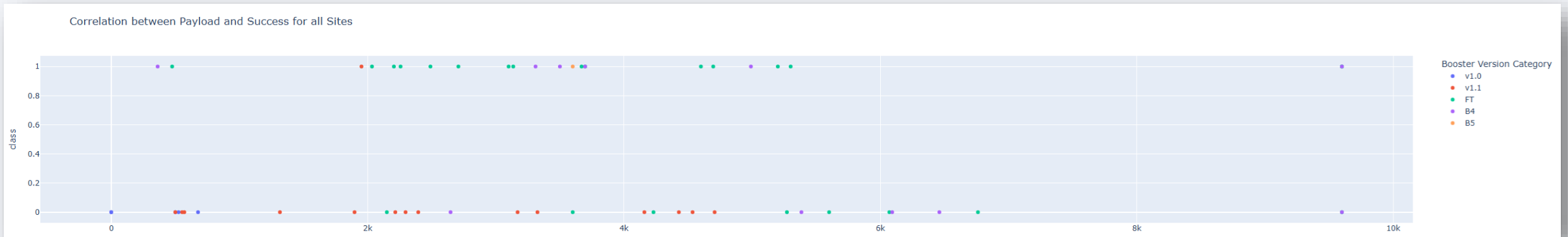


By filtering for a single site, the pie chart dynamically updates to show its specific performance. We can see the total success launches for site KSC-LC-39A Pie chart and a correlation between payload and success Scatter Plot. This view reveals that certain sites, like KSC LC 39A, have a particularly high success ratio.



# Payload vs. Outcome for Payloads between 2000-4000kg

---



For this medium payload range, the success rate is very high. This interactive tool allows for real-time analysis, showing that success rates remain high even with heavier payloads in later missions.

Section 5

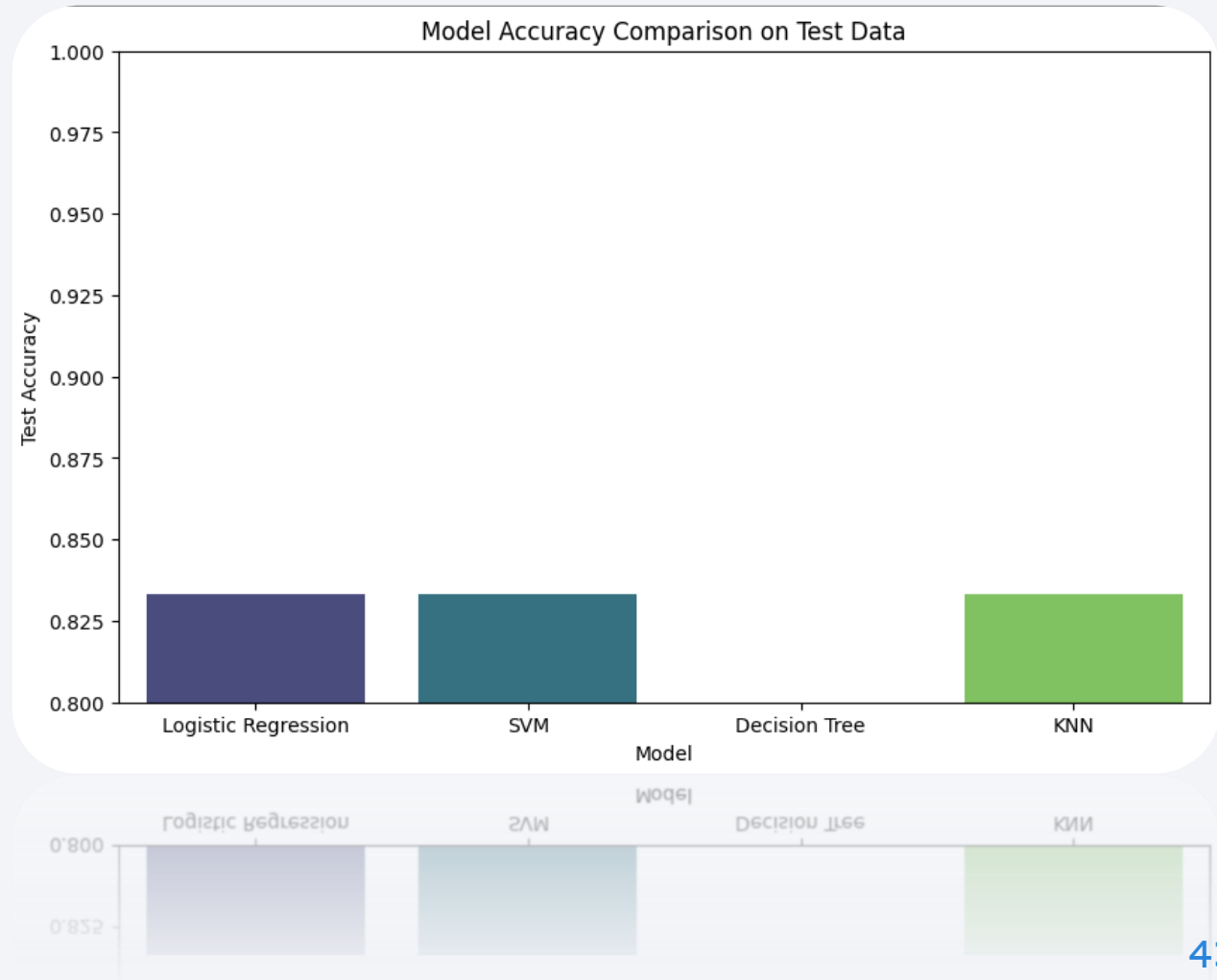
# Predictive Analysis (Classification)

# Classification Accuracy

## Accuracy Scores

- Logistic Regression: 83.33%
- Support Vector Machine (SVM): 83.33%
- Decision Tree: 77.78%
- K-Nearest Neighbors (KNN): 83.33%

All four machine learning models, after hyperparameter tuning, achieved the almost same accuracy of 83.33% except for decision tree model that achieves 77.78% on the test data. This indicates that the engineered features are strong predictors and that the problem is well-defined and solvable by multiple standard classifiers.



# Confusion Matrix

**Logistic Regression** model confusion matrix shows how the model's predictions on the test data compare to the actual outcomes.

**True Negatives (Top-Left): 3**

The model correctly predicted that the first stage did not land on 3 occasions.

**False Positives (Top-Right): 3**

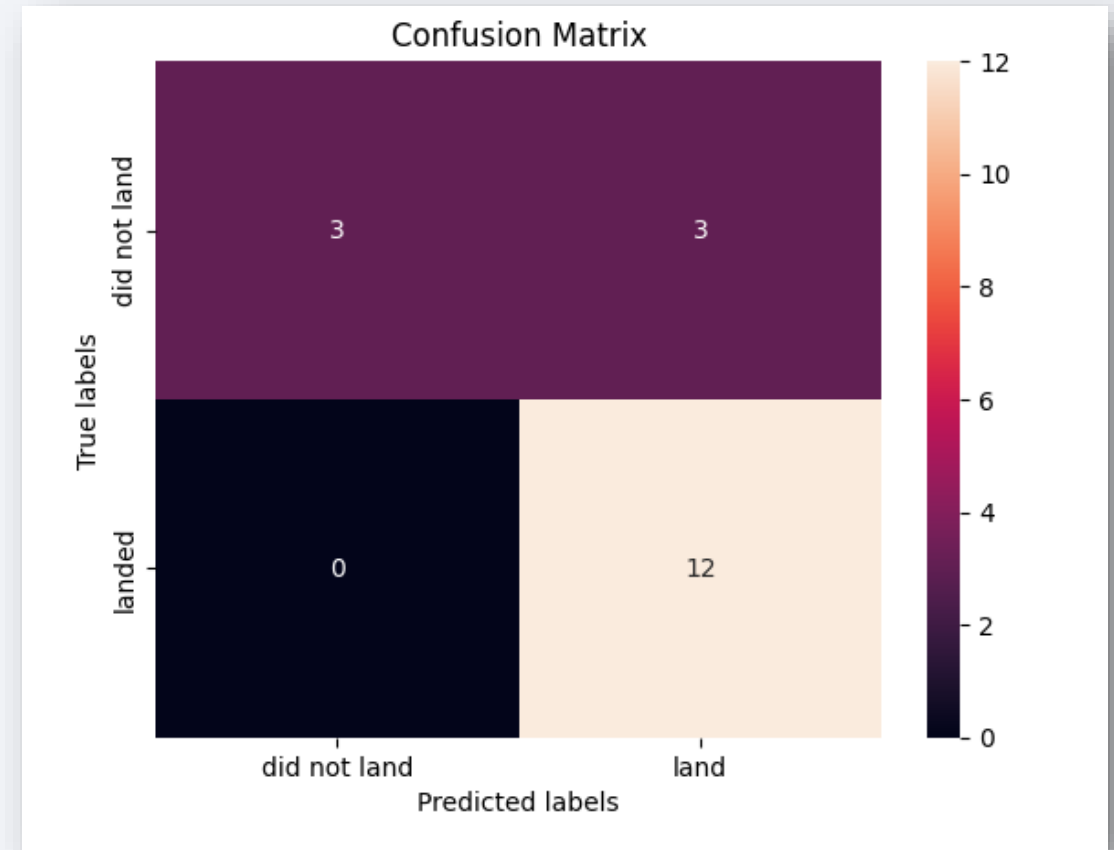
The model incorrectly predicted that the first stage landed when it actually did not land 3 times. This is the primary type of error the model makes.

**False Negatives (Bottom-Left): 0**

The model did not make this error. It never predicted that the first stage "did not land" when it actually did.

**True Positives (Bottom-Right): 12**

The model correctly predicted that the first stage landed on 12 occasions.



# Conclusions

---

- We identified that launch success is strongly correlated with the increasing FlightNumber, the specific Orbit type, and the booster Block version, demonstrating a clear learning curve and technological improvement over time.
- The machine learning analysis confirmed that first-stage landing success is predictable with a high degree of accuracy (83.33% across 3 different models), validating that the selected features are strong predictors.
- The resulting model serves as a valuable tool for competitive analysis. It enables a more accurate estimation of the true cost of a SpaceX launch by predicting the likelihood of first-stage reuse, which allows for more informed and competitive bidding.

# Appendix

---



<https://github.com/Ricardouchub/Data-Science-Capstone>



Thank you!

