IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Maverick Lising>
<December 1, 2025>

Outline

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

Executive Summary

- Summary of methodologies
- Summary of all results

Introduction

- Project background and context
- Problems you want to find answers

Section 1

Methodology

Methodology

Executive Summary

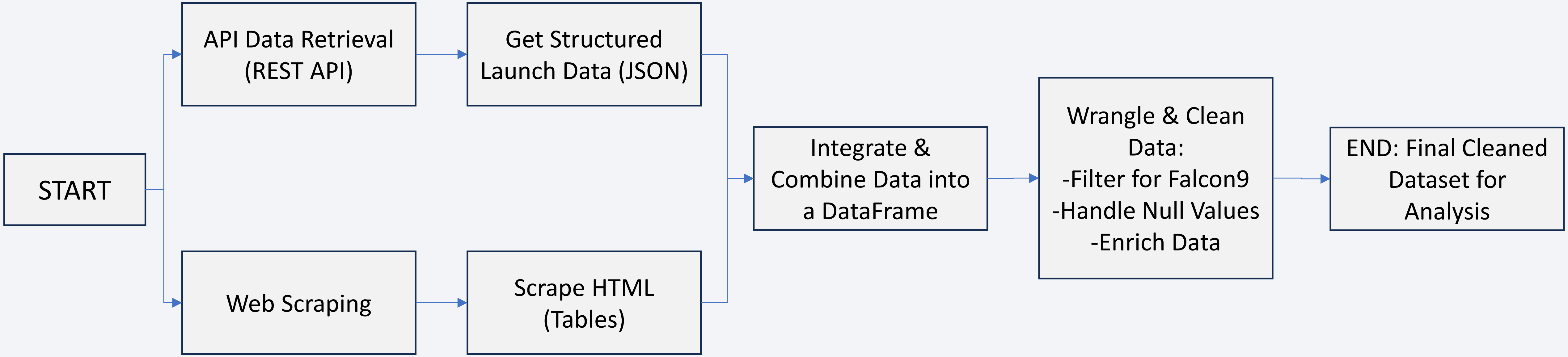
- Data collection methodology:
  - Describe how data was collected
- Perform data wrangling
  - Describe how data was processed
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

Data Collection Methodology

- The data for this project was gathered using a **two-pronged** approach to create a comprehensive dataset on SpaceX launches. The primary goal is to collect data to predict whether SpaceX will attempt to land a rocket. The two methods used are:
  - **API Data Retrieval**: The main source of data is the SpaceX REST API, which provides detailed, structured information about past launches.
  - **Web Scraping**: To supplement the API data, additional information, specifically about Falcon 9 launch records, was collected by scraping HTML tables from Wikipedia pages.

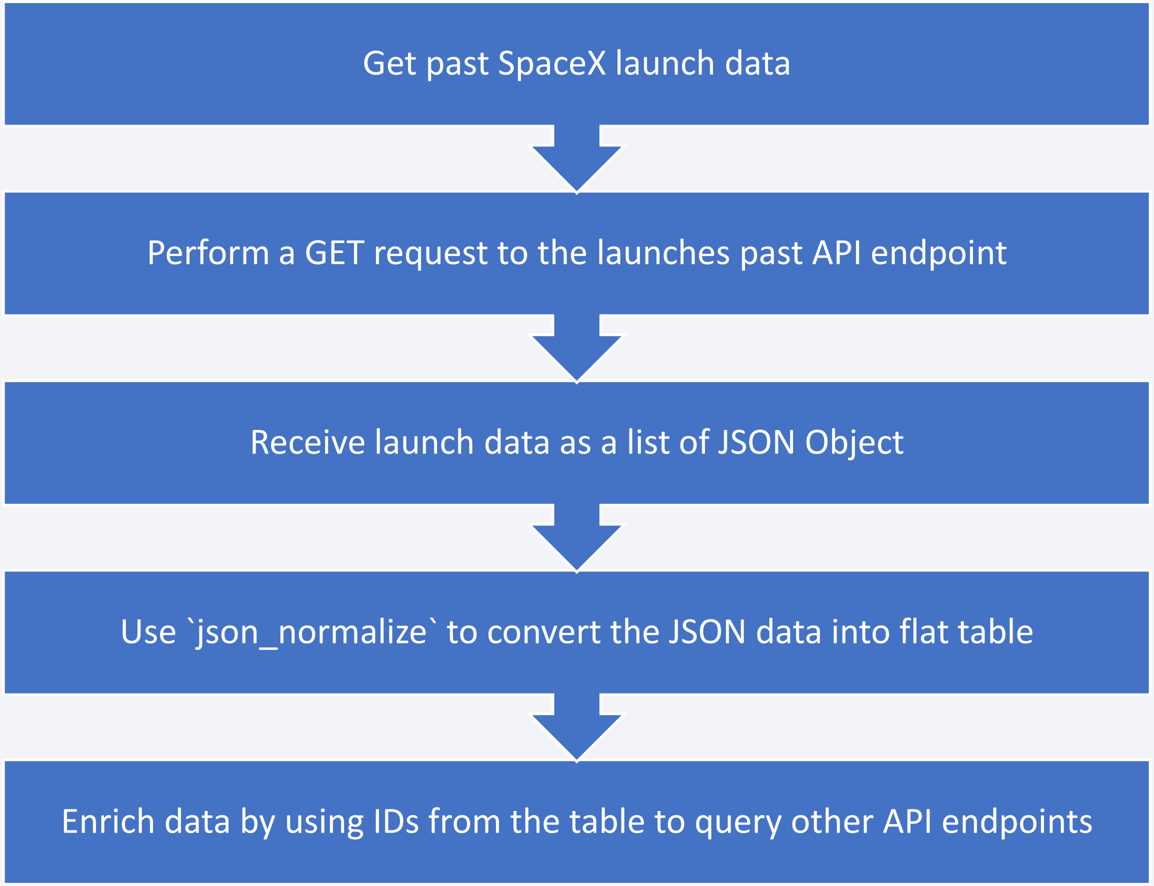
Data Collection


START
API Data Retrieval (REST API)
Get Structured Launch Data (JSON)
Web Scraping
Scrape HTML (Tables)
Integrate & Combine Data into a DataFrame
Wrangle & Clean Data:
-Filter for Falcon9
-Handle Null Values
-Enrich Data
END: Final Cleaned Dataset for Analysis

Data Collection – SpaceX API

- Present your data collection with SpaceX REST calls using key phrases and flowcharts

- GitHub URL



Get past SpaceX launch data

Perform a GET request to the launches past API endpoint

Receive launch data as a list of JSON Object

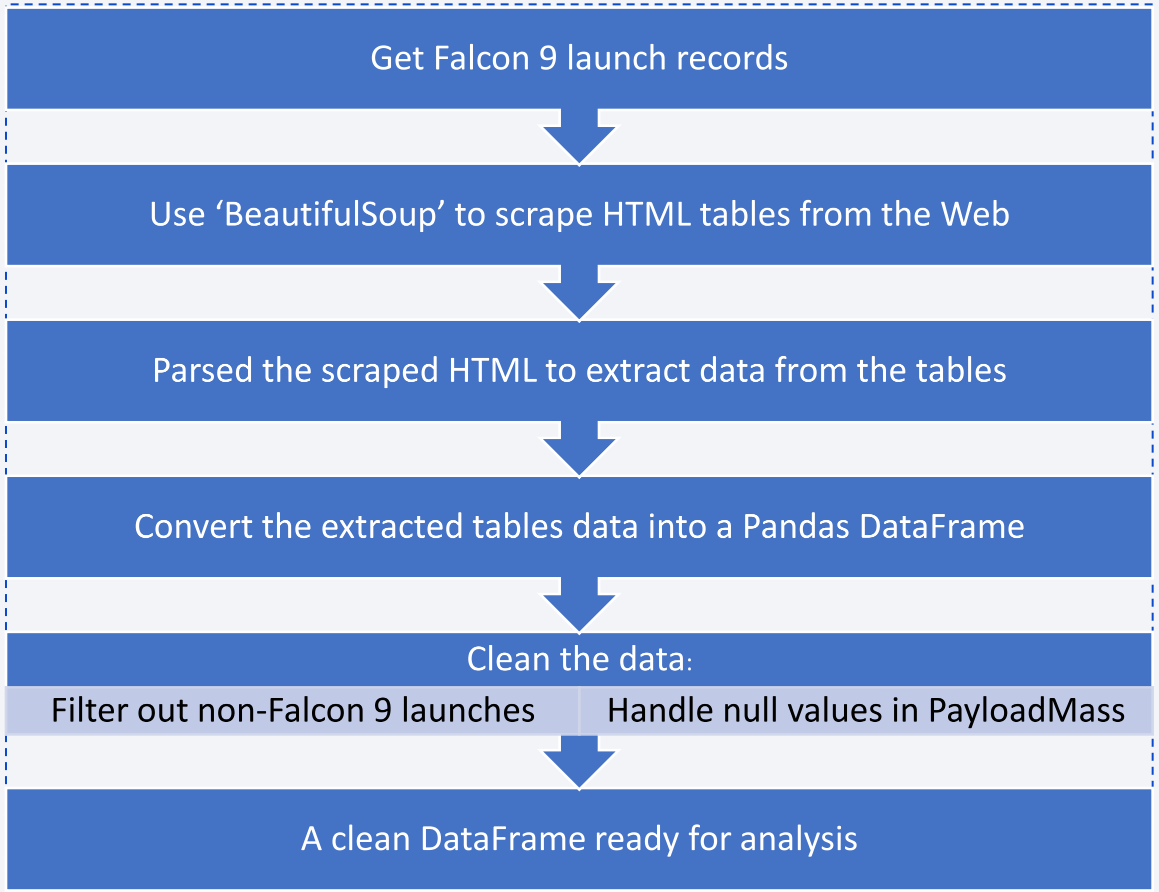
Use `json_normalize` to convert the JSON data into flat table

Enrich data by using IDs from the table to query other API endpoints

Data Collection - Scraping

- Present your web scraping process using key phrases and flowcharts

- GitHub URL



Get Falcon 9 launch records

Use ‘BeautifulSoup’ to scrape HTML tables from the Web

Parsed the scraped HTML to extract data from the tables

Convert the extracted tables data into a Pandas DataFrame

Clean the data:

Filter out non-Falcon 9 launches

Handle null values in PayloadMass

A clean DataFrame ready for analysis

Data Wrangling

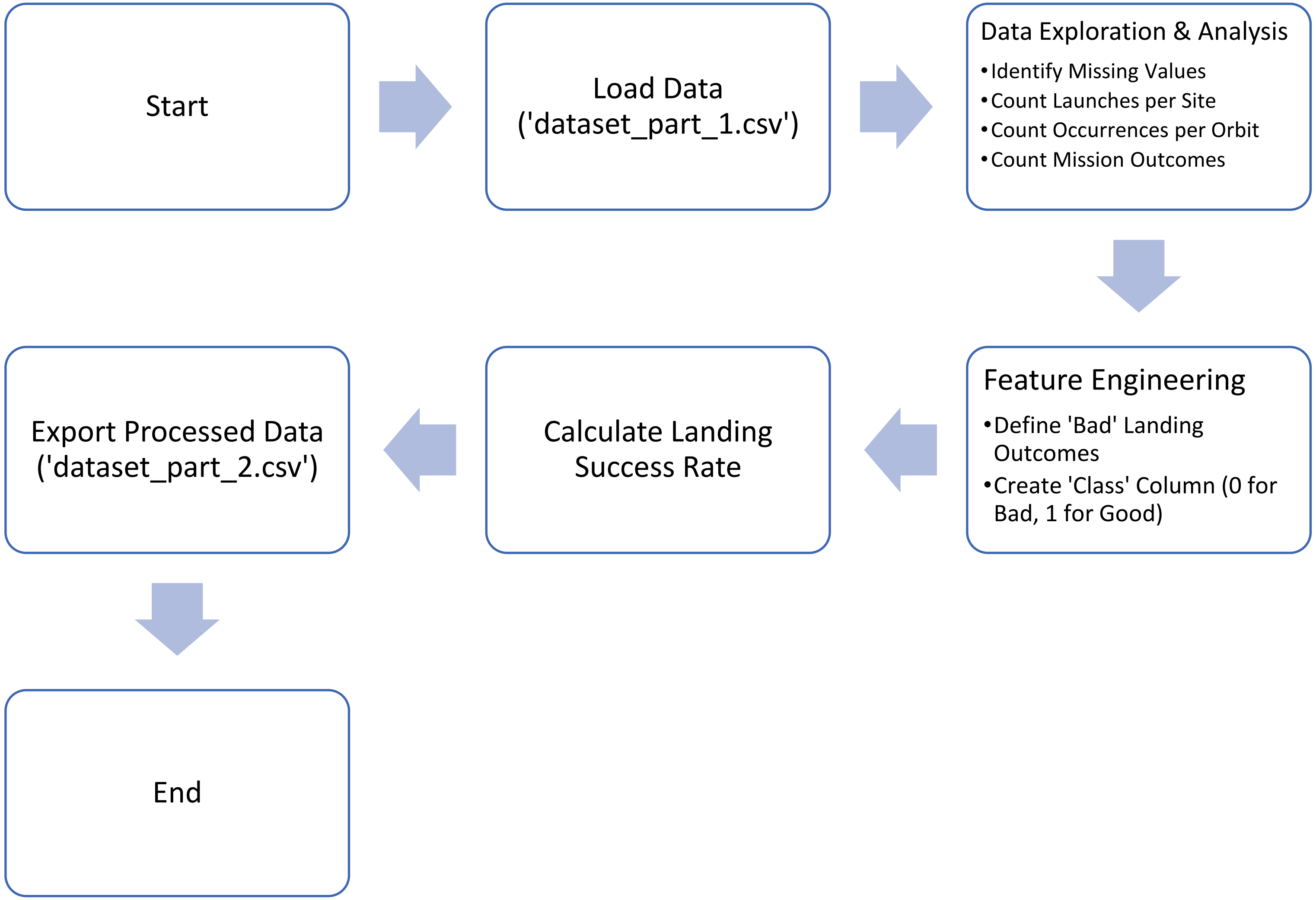
- Describe how data were processed - The data wrangling process transforms the initial SpaceX dataset into a format suitable for machine learning by cleaning the data and creating a target variable for prediction.
    - Data Loading: The process begins by loading the initial dataset from a CSV file into a pandas DataFrame.
    - Initial Data Assessment: The script performs an initial assessment by checking for missing values as a percentage of the total and identifying the data types for each column (e.g., numerical, categorical).
    - Exploratory Data Analysis (EDA): It explores the dataset by calculating the frequency of launches from each LaunchSite and the number of missions to each Orbit.
    - Outcome Analysis: The script analyzes the Outcome column to count the occurrences of different landing outcomes (e.g., True ASDS, False RTLS).

Data Wrangling

- Describe how data were processed (*continuation)*
  - Outcome Categorization: A set of "bad outcomes" is created to explicitly define all types of unsuccessful or failed landings.
  - Feature Engineering: A new binary classification column named Class is created. This column acts as the training label, where 1 represents a successful landing and 0 represents an unsuccessful one, based on whether the original Outcome was in the "bad outcomes" set.
  - Success Rate Calculation: The overall landing success rate is calculated by finding the mean of the newly created Class column.
  - Data Export: Finally, the processed DataFrame, now including the Class column, is saved to a new CSV file (dataset_part_2.csv) for the next stage of analysis.

Data Wrangling

- Data Wrangling Process Flow



Start

Load Data ('dataset_part_1.csv')

Data Exploration & Analysis

- Identify Missing Values
- Count Launches per Site
- Count Occurrences per Orbit
- Count Mission Outcomes

Feature Engineering

- Define 'Bad' Landing Outcomes
- Create 'Class' Column (0 for Bad, 1 for Good)

Calculate Landing Success Rate

Export Processed Data ('dataset_part_2.csv')

End

Data Wrangling

- GitHub URL

EDA with Data Visualization

- Summarize what charts were plotted and why you used those charts

Task No.	Chart Type	X-Axis	Y-Axis	Description & Purpose
0	Scatter Plot (catplot)	Flight Number	Payload Mass	This chart was used to explore the relationship between the cumulative number of flights, the mass of the payload, and the launch outcome (Class). It helps to see if more experience (higher flight number) and different payload masses affect the success rate.
1	Scatter Plot (catplot)	Flight Number	Launch Site	This plot visualizes if the success of a launch is related to its designated launch site over time. It helps identify if certain sites have better success records as more flights are launched.
2	Scatter Plot (catplot)	Payload Mass	Launch Site	Used to see if there's a relationship between the payload mass and the launch site. This can reveal if certain sites are specialized for lighter or heavier payloads and how that correlates with success.
3	Bar Chart (barplot)	Orbit	Success Rate	This chart compares the average success rate for each orbit type. A bar chart is effective here for directly comparing the performance of discrete categories (the different orbits).

EDA with Data Visualization

- Summarize what charts were plotted and why you used those charts

Task No.	Chart Type	X-Axis	Y-Axis	Description & Purpose
4	Scatter Plot (catplot)	Flight Number	Orbit	This plot helps to determine if the relationship between flight number and success is consistent across different orbit types. For example, it checks if success in a LEO orbit improves with more flights compared to a GTO orbit.
5	Scatter Plot (catplot)	Payload Mass	Orbit	This visualization explores the relationship between payload mass and launch success for various orbit types. It helps to see if successful landings are more common for certain payload weights in specific orbits.
6	Line Chart (lineplot)	Year	Success Rate	A line chart was used to show the trend of the average launch success rate over the years. This is an effective way to visualize performance changes over a continuous period.

EDA with Data Visualization

- Summarize what charts were plotted and why you used those charts

  Summary of Chart Choices:

  - Scatter Plots (catplot) were repeatedly used to visualize the relationship between two continuous variables (FlightNumber, PayloadMass) and a categorical variable (LaunchSite, Orbit), with the outcome (Class) encoded by color (hue). This is an excellent choice for identifying patterns, clusters, and correlations in the data.
  - A Bar Chart was used to compare the success rates of different categorical orbit types, providing a clear, at-a-glance comparison of performance.
  - A Line Chart was used to show the trend of success rates over time (yearly), which is the standard and most effective way to display time-series data.

EDA with Data Visualization

- GitHub URL

EDA with SQL

- Using bullet point format, summarize the SQL queries you performed
  - Task 1: Display Unique Launch Sites
    - Description: Retrieves the distinct names of all launch sites present in the space mission data.
    - SQL Script: SELECT DISTINCT Launch_Site FROM SPACEXTABLE;
  - Task 2: Filter Launch Sites by Prefix
    - Description: Displays the first 5 records for launch sites that have a name beginning with 'CCA'.
    - SQL Script: SELECT * FROM SPACEXTABLE WHERE Launch_Site LIKE 'CCA%' LIMIT 5;
  - Task 3: Calculate Total Payload Mass for NASA (CRS)
    - Description: Computes the sum of payload mass for all missions where the customer was "NASA (CRS)".
    - SQL Script: SELECT SUM(PAYLOAD_MASS__KG_) AS Total_Payload_Mass FROM SPACEXTABLE WHERE Customer = 'NASA (CRS)';
  - Task 4: Calculate Average Payload Mass for Booster Version F9 v1.1
    - Description: Determines the average payload mass for missions that used the "F9 v1.1" booster version.
    - SQL Script:SELECT AVG(PAYLOAD_MASS__KG_) AS Average_Payload_Mass FROM SPACEXTABLE WHERE Booster_Version = 'F9 v1.1';

EDA with SQL

- Using bullet point format, summarize the SQL queries you performed
  - Task 5: Find First Successful Ground Pad Landing Date
    - Description: Identifies the earliest date of a successful landing on a ground pad.
    - SQL Script: SELECT MIN(Date) AS First_Successful_Landing_Date FROM SPACEXTABLE WHERE Landing_Outcome = 'Success (ground pad)';
  - Task 6: List Boosters with Specific Landing and Payload Criteria
    - Description: Retrieves the names of boosters that had a successful drone ship landing and carried a payload mass between 4,000 and 6,000 kg.
    - SQL Script: SELECT Booster_Version FROM SPACEXTABLE WHERE Landing_Outcome = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000;
  - Task 7: Count Successful and Failed Mission Outcomes
    - Description: Groups the missions by their outcome (e.g., success, failure) and counts the total number in each category.
    - SQL Script: SELECT Mission_Outcome, COUNT(*) AS Total_Count FROM SPACEXTABLE GROUP BY Mission_Outcome;

EDA with SQL

- Using bullet point format, summarize the SQL queries you performed
    - Task 8: Identify Boosters with Maximum Payload
        - Description: Lists all booster versions that have carried the maximum payload mass, using a subquery.
        - SQL Script: SELECT Booster_Version FROM SPACEXTABLE WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE);
    - Task 9: List Failed Drone Ship Landings in 2015
        - Description: Shows the month, landing outcome, booster version, and launch site for all failed drone ship landings that occurred in 2015.
        - SQL Script: 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)';
    - Task 10: Rank Landing Outcomes by Date Range
        - Description: Counts and ranks the landing outcomes between June 4, 2010, and March 20, 2017, in descending order of frequency.
        - SQL Script: SELECT Landing_Outcome, COUNT(*) AS Outcome_Count FROM SPACEXTABLE WHERE Date BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY Landing_Outcome ORDER BY Outcome_Count DESC;

EDA with SQL

- GitHub URL

Build an Interactive Map with Folium

- Summarize what map objects such as markers, circles, lines, etc. you created and added to a folium map

Object Type	Description
folium.Circle	A circular marker with a specified radius and color.
folium.map.Marker	A point marker on the map. It was used with two types of icons:
	- DivIcon: A custom HTML-based icon, used to display text labels directly on the map.
	- folium.Icon: A standard, colored pin marker.
MarkerCluster	A plugin that groups multiple markers from the same area into a single cluster, which separates as you zoom in.
MousePosition	A plugin that displays the mouse cursor's geographic coordinates (latitude and longitude) on the map.
folium.PolyLine	A line drawn on the map to connect a sequence of coordinates.

Build an Interactive Map with Folium

- Explain why you added those objects

1. To Mark All Launch Sites (Task 1)
The initial goal was to visualize the geographic locations of all SpaceX launch sites.

  - folium.Circle: A circle was added at the coordinates of each launch site to create a visually distinct, highlighted area for each location.
  - folium.map.Marker with DivIcon: A text label with the name of the launch site (e.g., CCAFS LC-40) was placed at each site's coordinates. This provides a clear, persistent identifier for each marked circle.

2. To Show Launch Success/Failure Rates (Task 2)
The next step was to visualize the outcome of every launch from each site.

  - folium.Marker with folium.Icon: A marker was added for every single launch record. The marker's color was set to green for a successful launch and red for a failed one. This provides an immediate visual cue for the outcome of each mission.
  - MarkerCluster: Since many launches originate from the exact same coordinates, all the individual success/failure markers were added to a MarkerCluster. This prevents the map from becoming cluttered. The cluster shows the total number of launches from a site, and zooming in reveals the individual red and green markers, making it easy to see the success rate per site.

Build an Interactive Map with Folium

- Explain why you added those objects

3. To Analyze Proximity to Infrastructure (Task 3)
The final task was to measure the distance from a launch site to nearby points of interest like coastlines, highways, and cities.

  - MousePosition: This tool was added to allow the user to interactively find the coordinates of any point on the map by simply moving the mouse over it. This was necessary to identify the locations of nearby infrastructure.
  - folium.PolyLine: A line was drawn connecting the launch site to a point of interest (e.g., the closest coastline). This visually represents the distance being measured.
  - folium.map.Marker with DivIcon: At the location of the point of interest (e.g., the coastline), a text marker was added to display its name and the calculated distance from the launch site (e.g., "Coastline 0.96 KM"). This makes the results of the distance analysis clear and easy to read.

Build an Interactive Map with Folium

- GitHub URL

Build a Dashboard with Plotly Dash

- Summarize what plots/graphs and interactions you have added to a dashboard

The Dash application is designed to visualize SpaceX launch data. It includes two main interactive components and two plots:

- Interactions:
  - Launch Site Dropdown: Allows users to filter the dashboard to show data for all launch sites or a single, specific launch site.
  - Payload Range Slider: Enables users to filter the data based on a selected range of payload mass in kilograms.
- Plots/Graphs:
  - Success Pie Chart: A pie chart that shows the distribution of successful launches.
  - Payload vs. Outcome Scatter Chart: A scatter plot that visualizes the relationship between payload mass and launch success, segmented by booster version.

Build a Dashboard with Plotly Dash

- Explain why you added those plots and interactions

Plots and Graphs

- Success Pie Chart (success-pie-chart)
  - Purpose: This chart's goal is to visualize the success rate of launches.
  - Functionality: Its content changes based on the "Launch Site Dropdown" selection.
    - If "All Sites" is selected, the pie chart displays the proportion of total successful launches contributed by each launch site.
    - If a specific launch site is selected, the chart switches to show the proportion of successful launches (class 1) versus failed launches (class 0) for only that site.
- Payload vs. Outcome Scatter Chart (success-payload-scatter-chart)
  - Purpose: This chart is used to investigate the relationship between the mass of the payload and the success of the launch.
  - Functionality: The plot is controlled by both the "Launch Site Dropdown" and the "Payload Range Slider."
    - The x-axis represents "Payload Mass (kg)."
    - The y-axis represents the launch outcome, where 1 is a success and 0 is a failure.
    - The points on the graph are colored by the "Booster Version Category," allowing users to see if certain booster versions perform better with different payload weights.

Build a Dashboard with Plotly Dash

- Explain why you added those plots and interactions

Interactions

- Launch Site Dropdown (site-dropdown)
  - Purpose: This is the primary filter for the dashboard.
  - Functionality: It's a searchable dropdown menu that contains "All Sites" as the default option, along with all unique launch sites from the dataset. Selecting an option from this dropdown updates both the pie chart and the scatter plot to reflect data for the chosen site(s).
- Payload Range Slider (payload-slider)
  - Purpose: This slider allows for more granular analysis of the payload's impact on launch outcomes.
  - Functionality: Users can drag the handles to select a minimum and maximum payload mass. This action filters the data shown in the "Payload vs. Outcome Scatter Chart," which will then only display launches that fall within the selected payload range. This slider does not affect the pie chart.

Build a Dashboard with Plotly Dash

- GitHub URL

Predictive Analysis (Classification)

- Summarize how you built, evaluated, improved, and found the best performing classification model

1. Building the Foundation (Data Preparation):
    - Data Loading: The features (X) and the target variable 'Class' (Y) were loaded from two separate CSV files.
    - Data Standardization: The feature data was standardized using StandardScaler. This step ensures that all features are on the same scale, which is crucial for distance-based algorithms like SVM and for the regularization in Logistic Regression.
    - Data Splitting: The dataset was split into a training set (80%) and a testing set (20%) to ensure the final model could be evaluated on unseen data.
2. Improving the Models (Hyperparameter Tuning):
    - The core of the improvement process was using GridSearchCV with 10-fold cross-validation. This technique systematically tests various model configurations to find the optimal set of hyperparameters.
    - Four different classification algorithms were tuned:
        - Logistic Regression: Tuned for parameters like C (regularization strength) and solver.
        - Support Vector Machine (SVM): Tuned for the kernel, C, and gamma parameters.
        - Decision Tree: Tuned for a wide range of parameters, including criterion, max_depth, and min_samples_leaf.
        - K-Nearest Neighbors (KNN): Tuned for the number of n_neighbors and the distance metric p.

Predictive Analysis (Classification)

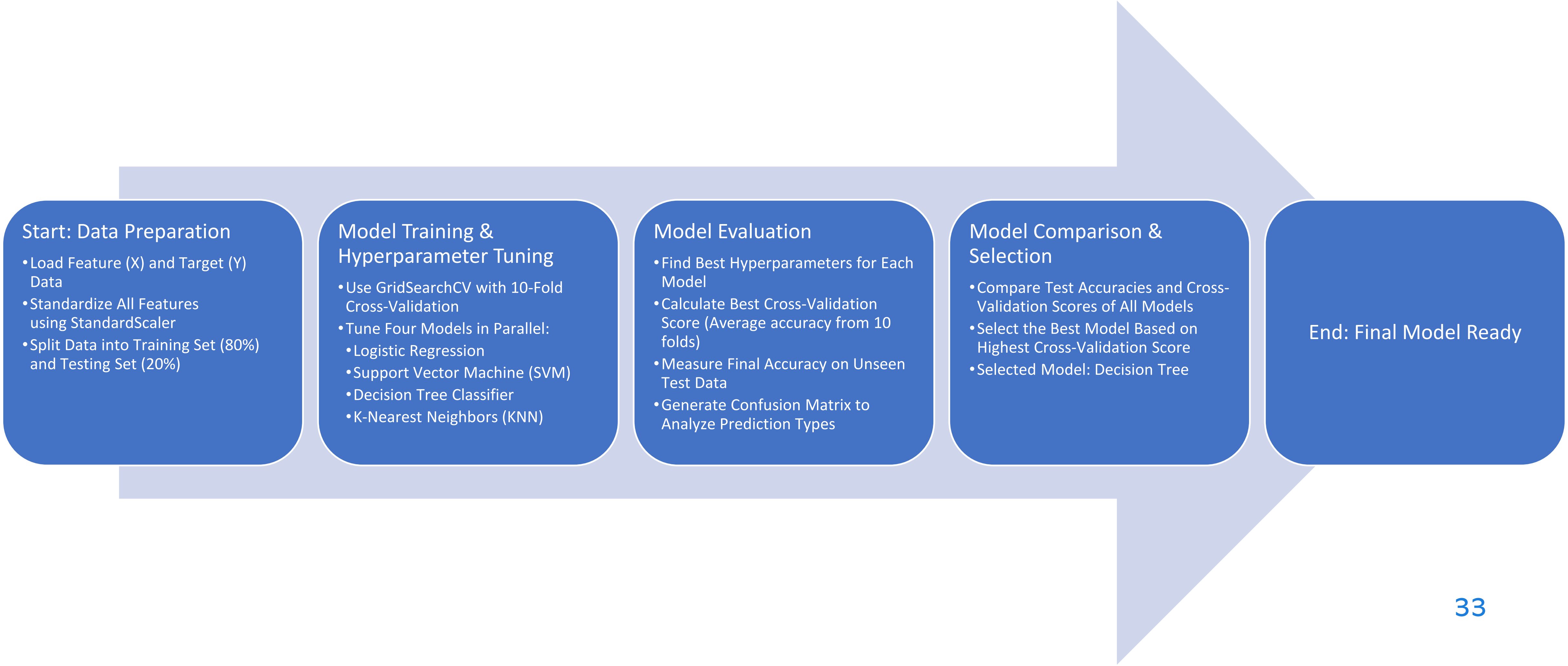
- Summarize how you built, evaluated, improved, and found the best performing classification model

3. Evaluating Performance:
   - Cross-Validation Score: During tuning, GridSearchCV provides a best_score_, which is the average accuracy across the 10 cross-validation folds on the training data. This score is a robust indicator of a model's performance.
   - Test Accuracy: After tuning, each model's final performance was measured by its accuracy on the separate, unseen test data.
   - Confusion Matrix: A confusion matrix was generated for each model to provide a more detailed breakdown of its predictions, showing correct predictions versus incorrect ones (false positives and false negatives).

4. Finding the Best Performing Model:
   - The final step was to compare the performance of all four tuned models. While all models achieved the same accuracy of 83.33% on the test data, the Decision Tree model was identified as the best performer because it achieved the highest cross-validation score during the tuning phase, indicating it was the most consistently accurate model on different subsets of the training data.

Predictive Analysis (Classification)

- You need present your model development process using key phrases and flowchart



Start: Data Preparation

- Load Feature (X) and Target (Y) Data
- Standardize All Features using StandardScaler
- Split Data into Training Set (80%) and Testing Set (20%)

Model Training & Hyperparameter Tuning

- Use GridSearchCV with 10-Fold Cross-Validation
- Tune Four Models in Parallel:
  - Logistic Regression
  - Support Vector Machine (SVM)
  - Decision Tree Classifier
  - K-Nearest Neighbors (KNN)

Model Evaluation

- Find Best Hyperparameters for Each Model
- Calculate Best Cross-Validation Score (Average accuracy from 10 folds)
- Measure Final Accuracy on Unseen Test Data
- Generate Confusion Matrix to Analyze Prediction Types

Model Comparison & Selection

- Compare Test Accuracies and Cross-Validation Scores of All Models
- Select the Best Model Based on Highest Cross-Validation Score
- Selected Model: Decision Tree

End: Final Model Ready

Predictive Analysis (Classification)

- GitHub URL

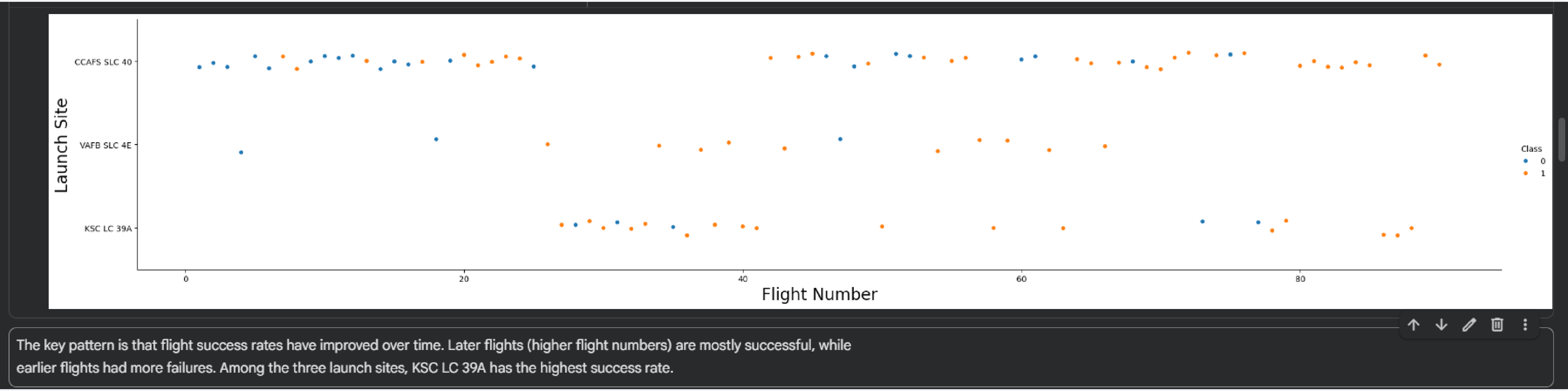
Results

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

Section 2

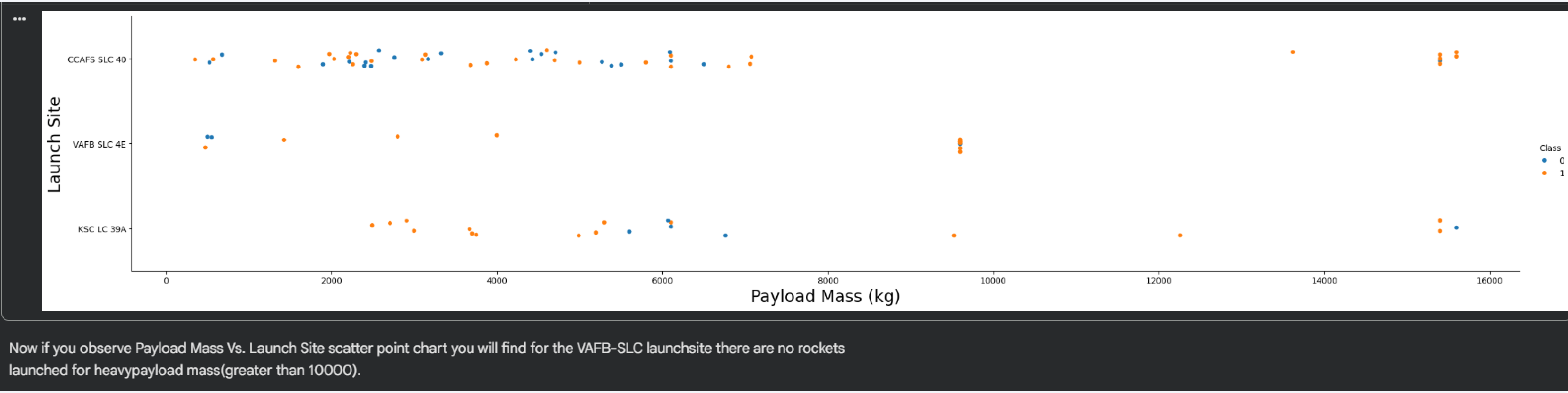
Insights drawn from EDA

Flight Number vs. Launch Site



The key pattern is that flight success rates have improved over time. Later flights (higher flight numbers) are mostly successful, while earlier flights had more failures. Among the three launch sites, KSC LC 39A has the highest success rate.

Payload vs. Launch Site



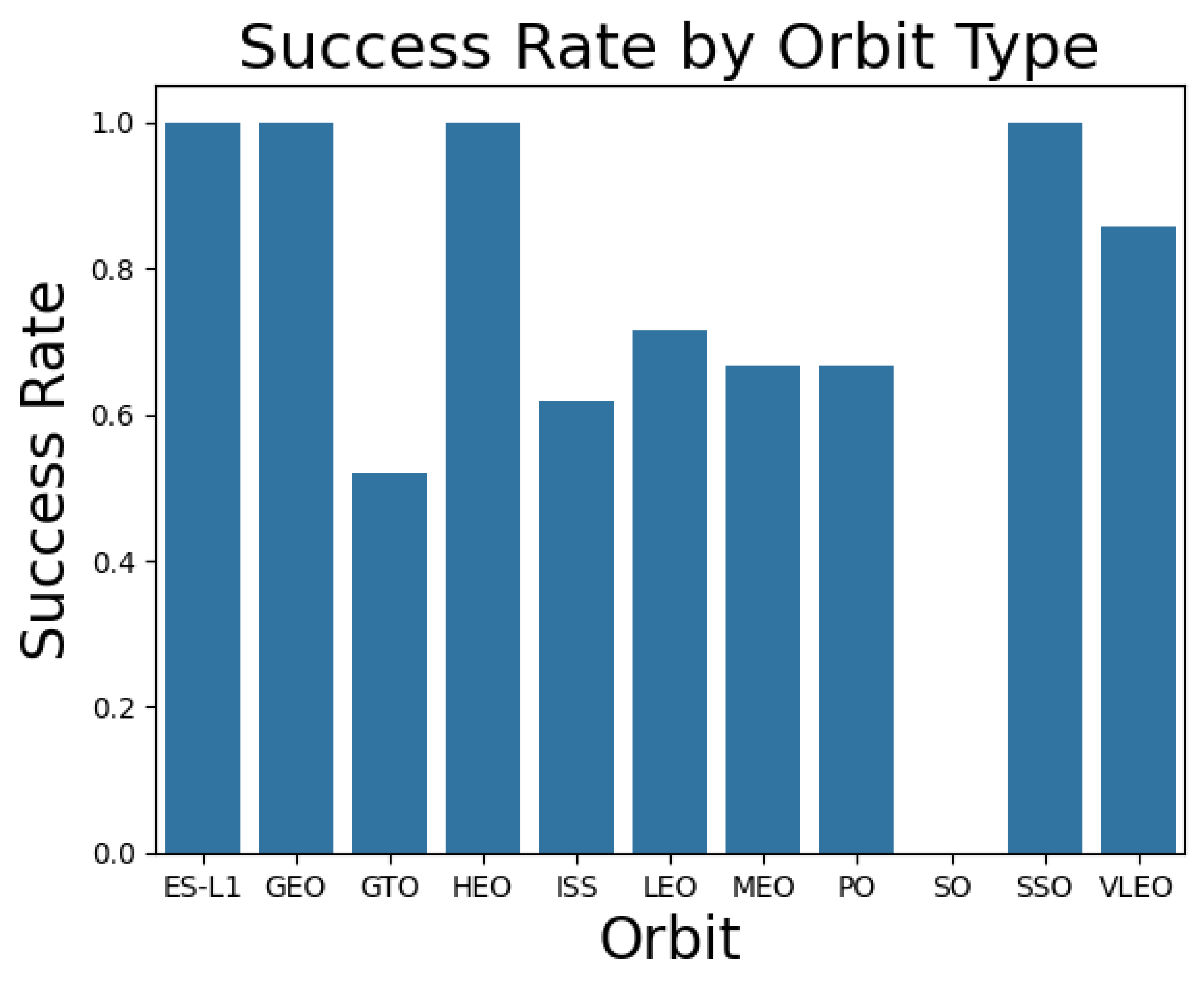
Now if you observe Payload Mass Vs. Launch Site scatter point chart you will find for the VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000).

Success Rate vs. Orbit Type

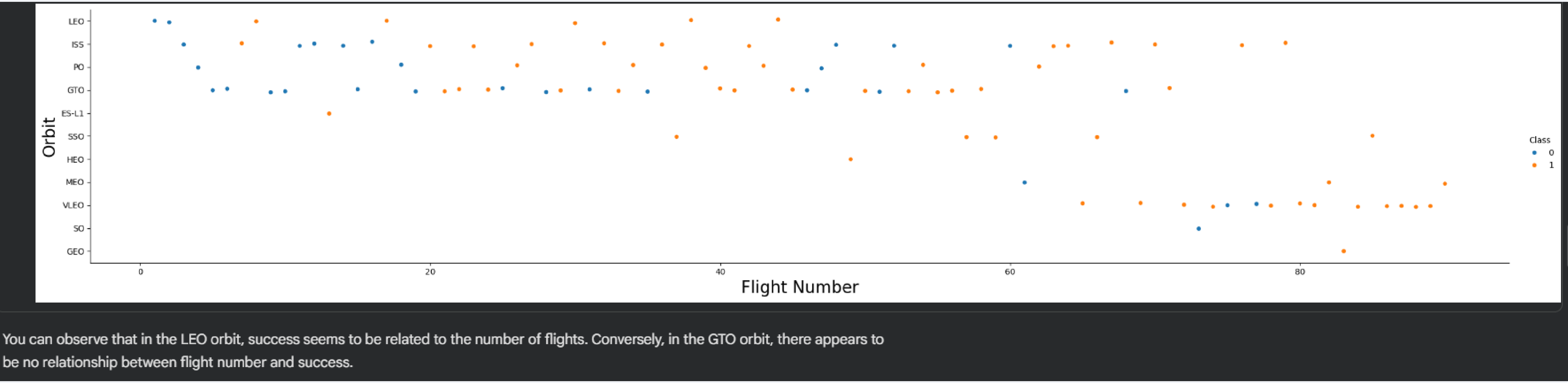
Based on the "Success Rate by Orbit Type" bar chart, the following orbits have the highest success rates:

- 100% Success Rate: The orbits ES-L1, GEO, HEO, and SSO all show a perfect success rate of 1.0.
- High Success Rate: The VLEO orbit also has a high success rate, approximately 0.86 (or 86%).

In contrast, the GTO orbit has the lowest success rate among those shown, at just over 50%. The SO orbit appears to have no data.

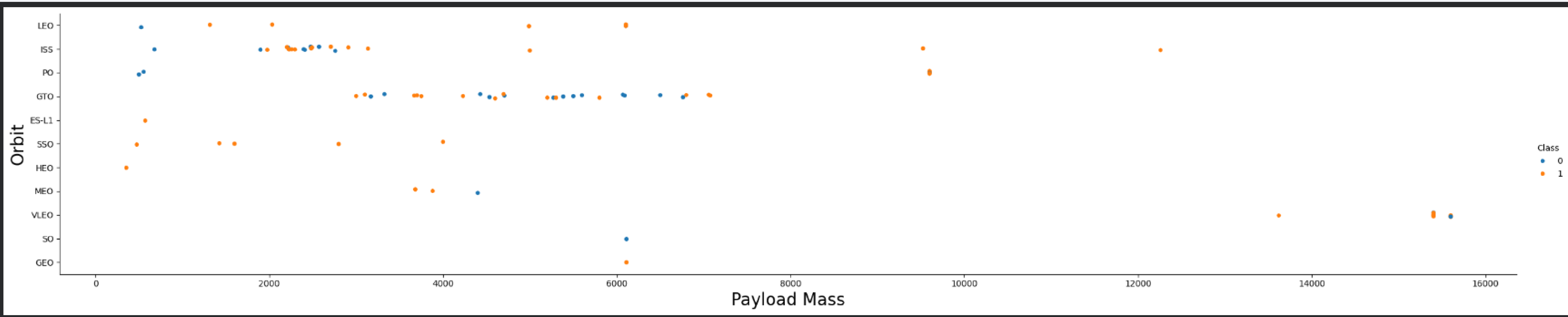


Flight Number vs. Orbit Type



You can observe that in the LEO orbit, success seems to be related to the number of flights. Conversely, in the GTO orbit, there appears to be no relationship between flight number and success.

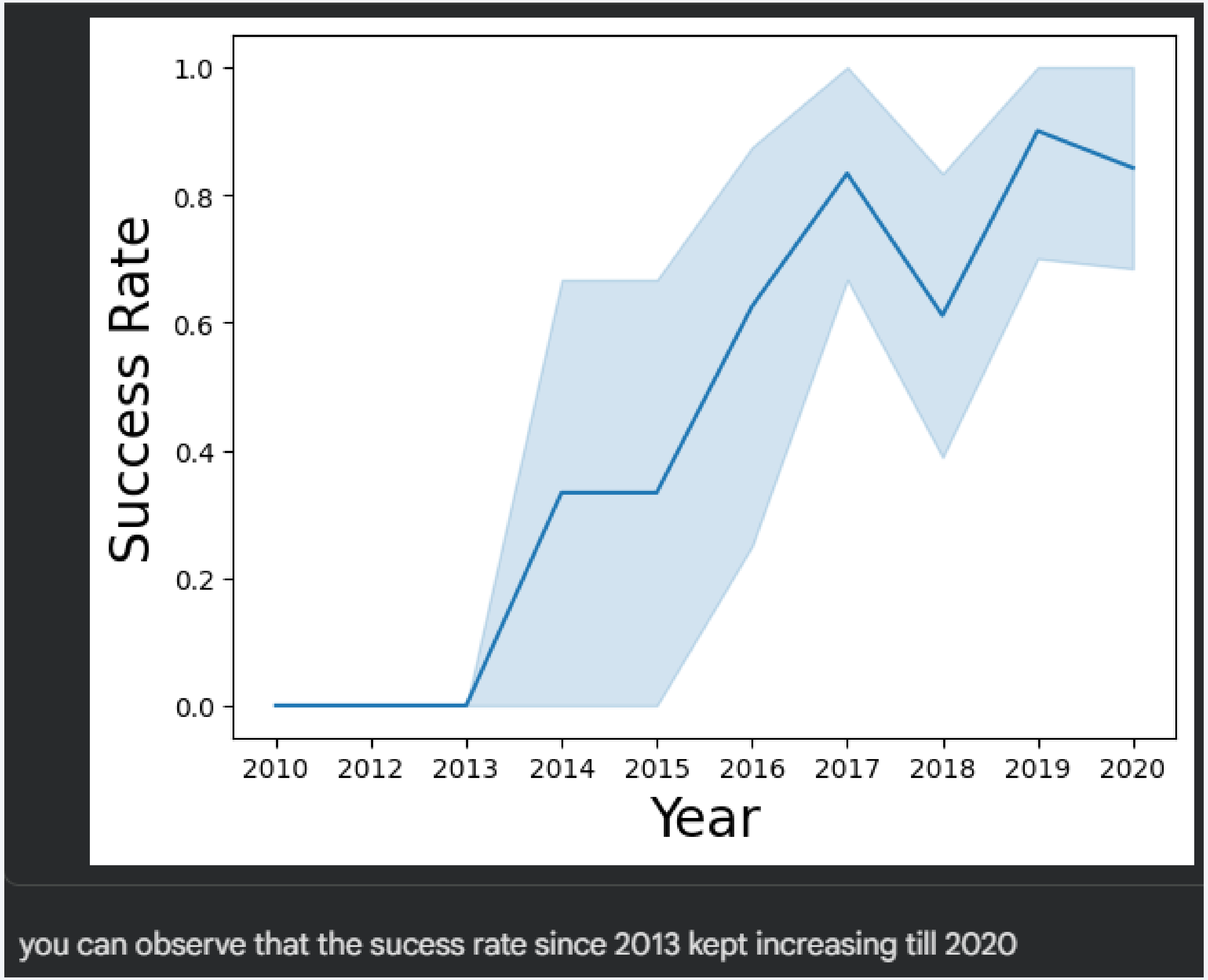
Payload vs. Orbit Type



ith heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.

However, for GTO, it's difficult to distinguish between successful and unsuccessful landings as both outcomes are present.

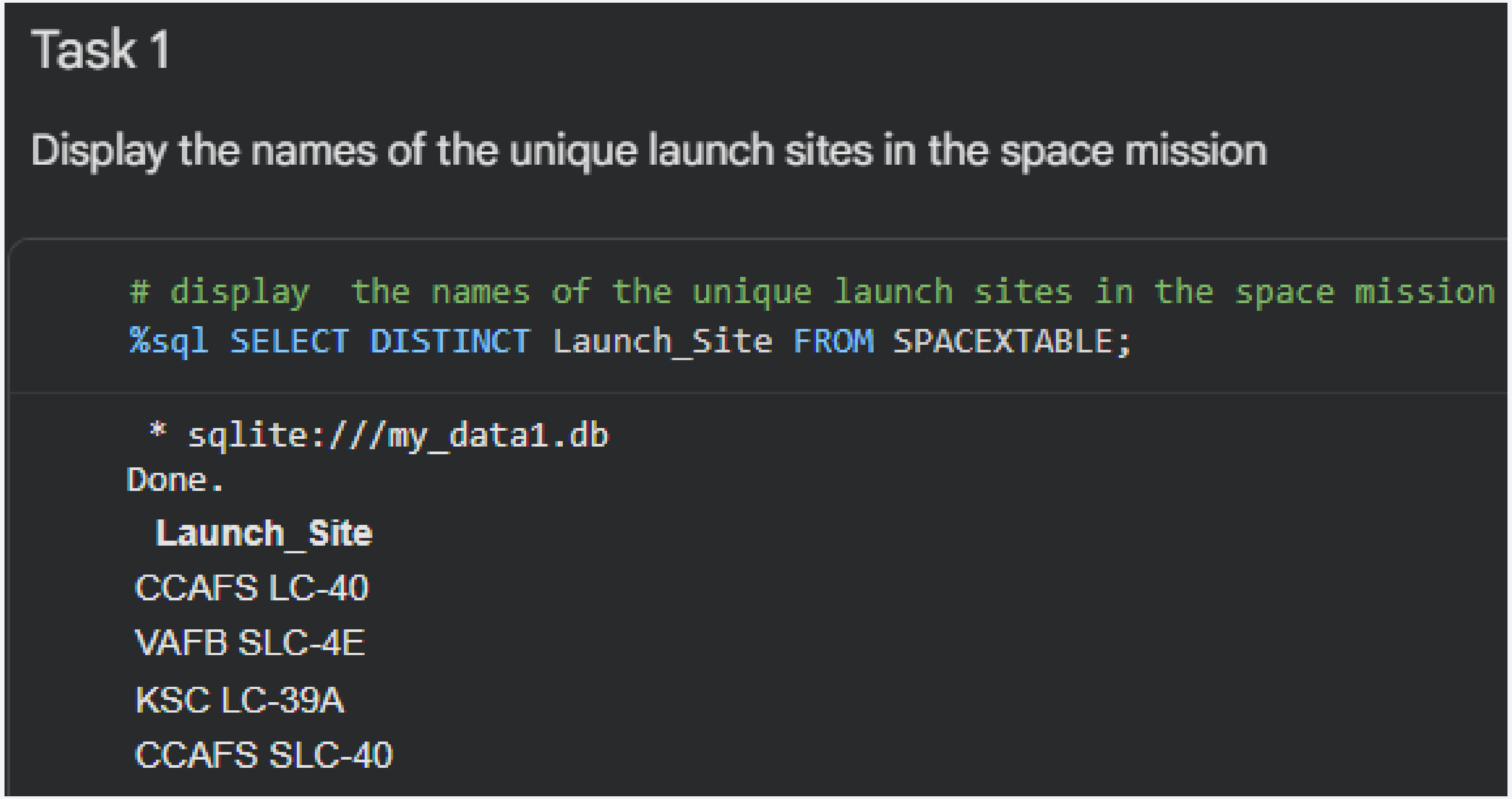
Launch Success Yearly Trend



you can observe that the sucess rate since 2013 kept increasing till 2020

All Launch Site Names

- Find the names of the unique launch sites
- Present your query result with a short explanation here



Task 1

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

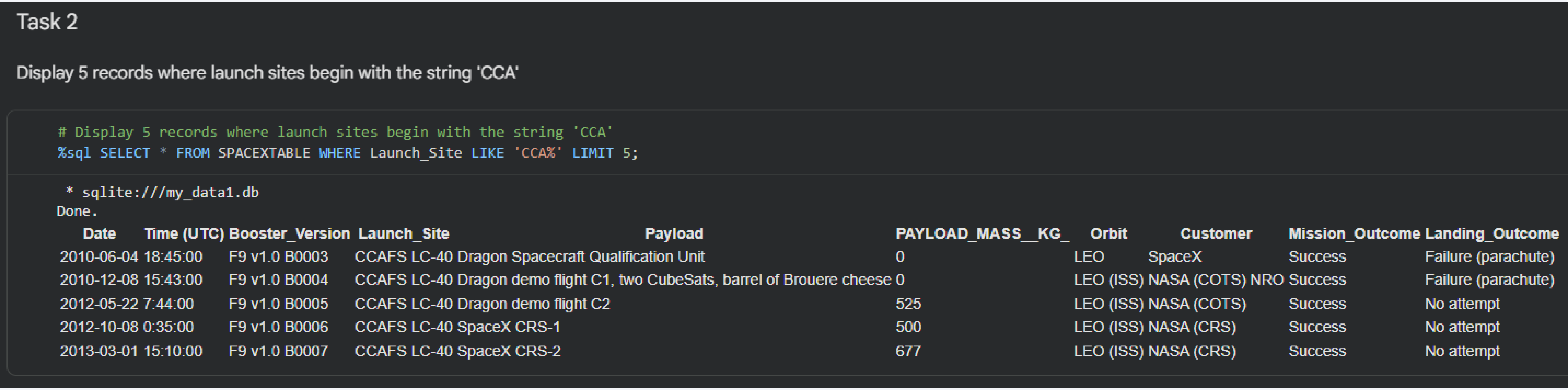
```
# display  the names of the unique launch sites in the space mission
%sql SELECT DISTINCT Launch_Site FROM SPACEXTABLE;
```

```
 * sqlite:///my_data1.db
Done.
 Launch_Site
CCAFS LC-40
VAFB SLC-4E
KSC LC-39A
CCAFS SLC-40
```

From the table `SPACEXTABLE`, we have obtained all the unique Launch Sites values namely: CCAFS LC-40, VAFB SLC-4E, KSC LC-39A and CCAFS SLC-40

Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA`
- Present your query result with a short explanation here



Task 2

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

```
# Display 5 records where launch sites begin with the string 'CCA'
%sql SELECT * FROM SPACEXTABLE WHERE Launch_Site LIKE 'CCA%' LIMIT 5;
```

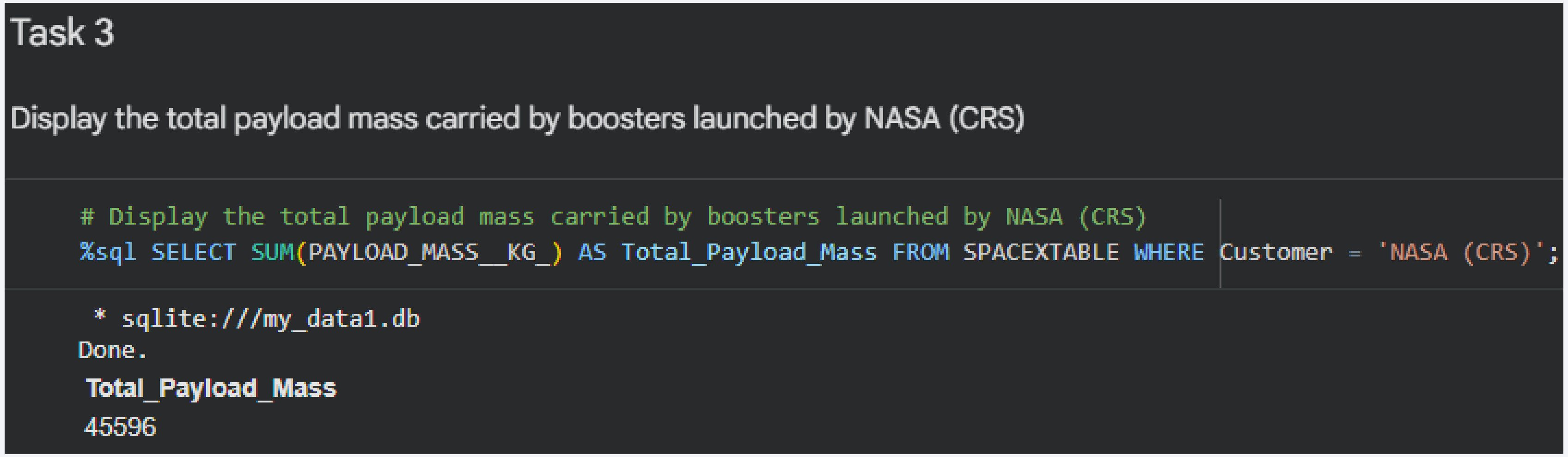
```
 * sqlite:///my_data1.db
Done.
```

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

The SQL query was designed to display the first 5 records from the database where the Launch_Site character name begins with "CCA"

Total Payload Mass

- Calculate the total payload carried by boosters from NASA
- Present your query result with a short explanation here



Task 3

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

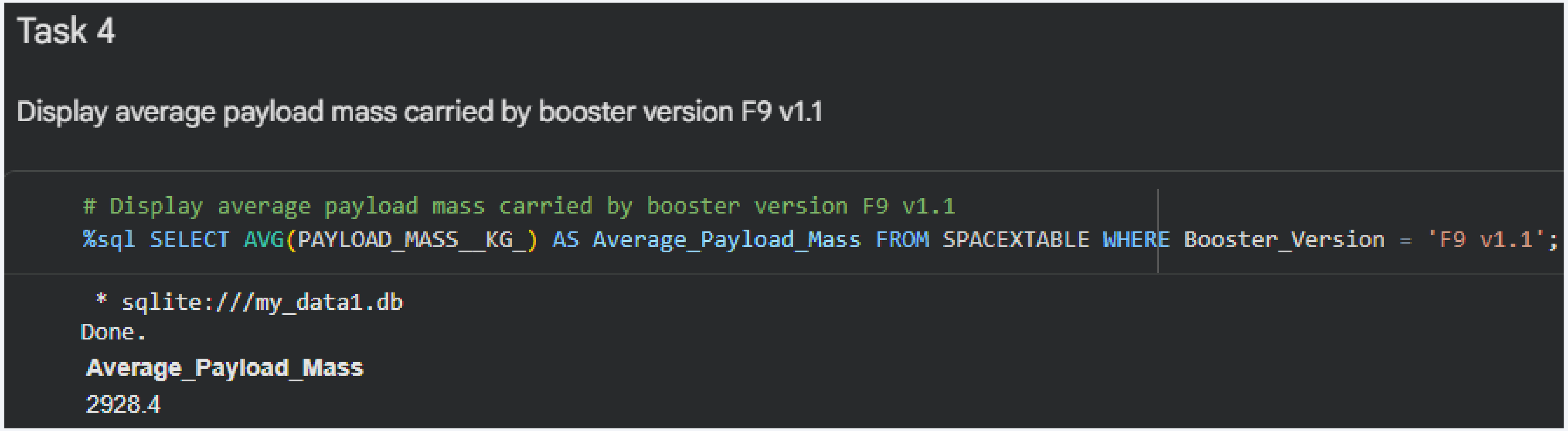
```
# Display the total payload mass carried by boosters launched by NASA (CRS)
%sql SELECT SUM(PAYLOAD_MASS__KG_) AS Total_Payload_Mass FROM SPACEXTABLE WHERE Customer = 'NASA (CRS)';
```

```
 * sqlite:///my_data1.db
Done.
 Total_Payload_Mass
 45596
```

This query was executed to identify all successful landings on a drone ship where the payload mass was between 2,000 kg and 7,500 kg.
The resulting table shows 9 missions that fit these criteria. All listed launches had a Landing_Outcome of 'Success (drone ship)', and their payload masses fall within the specified range, from 1,952 kg to 7,000 kg.

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1
- Present your query result with a short explanation here



Task 4

Display 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 Average_Payload_Mass FROM SPACEXTABLE WHERE Booster_Version = 'F9 v1.1';
```

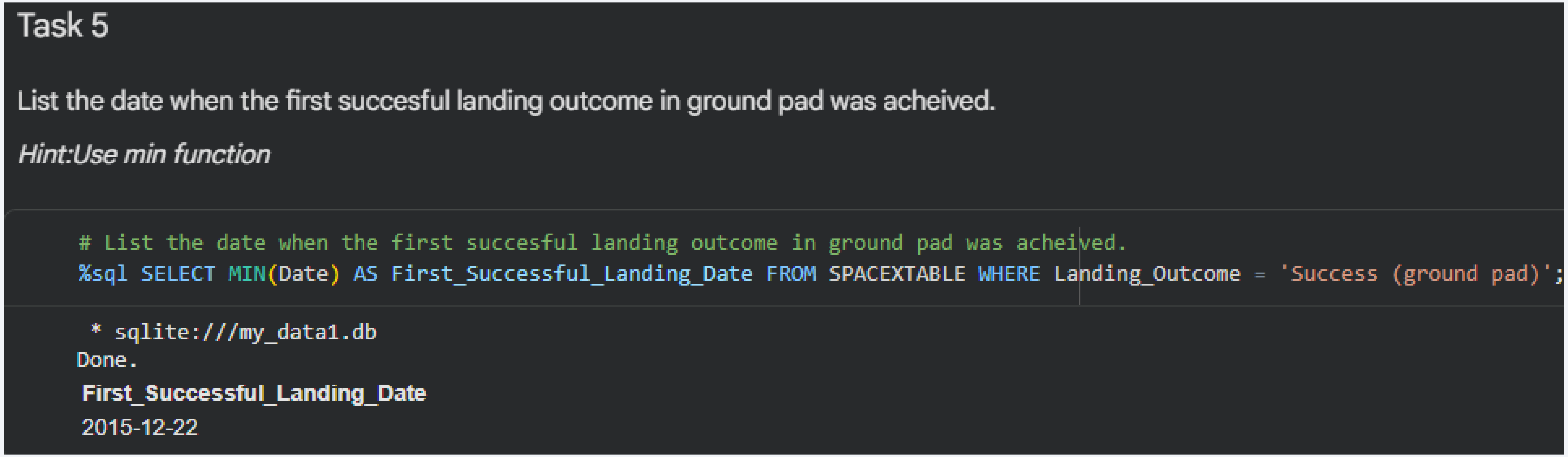
```
 * sqlite:///my_data1.db
Done.
```

Average_Payload_Mass
2928.4

This SQL query was executed to calculate the average payload mass for all launches that used the F9 v1.1 booster version.
The result shows that the average payload mass carried by this specific booster version was 2928.4 kg.

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad
- Present your query result with a short explanation here



Task 5

List the date when the first succesful landing outcome in ground pad was acheived.

Hint:Use min function

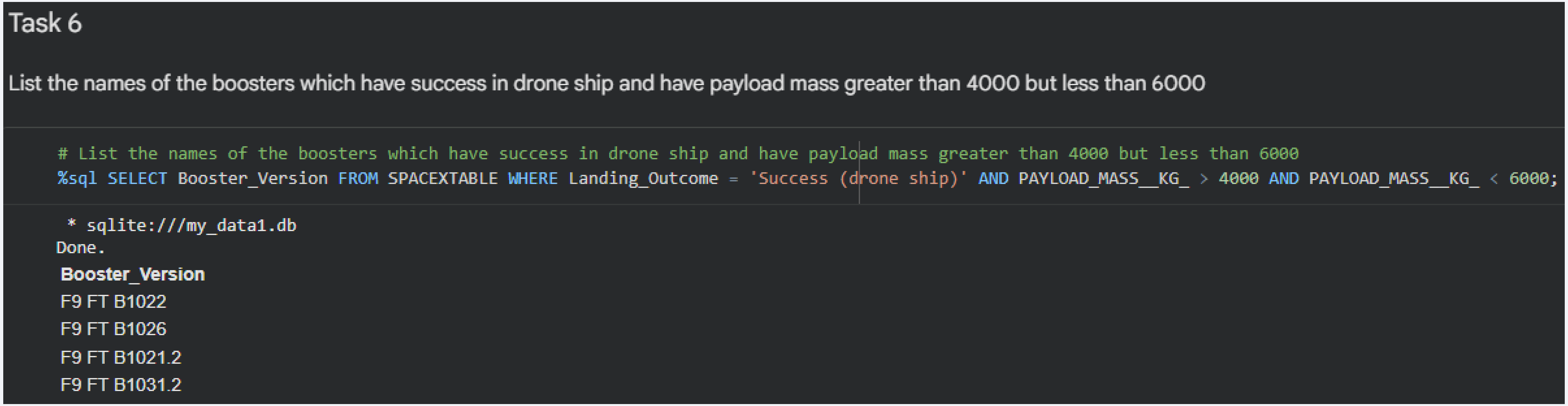
```
# List the date when the first succesful landing outcome in ground pad was acheived.
%sql SELECT MIN(Date) AS First_Successful_Landing_Date FROM SPACEXTABLE WHERE Landing_Outcome = 'Success (ground pad)';
```

```
 * sqlite:///my_data1.db
Done.
First_Successful_Landing_Date
2015-12-22
```

This SQL query was executed to find the earliest date of a successful landing on a ground pad. It filters all launches to find those with a Landing_Outcome of 'Success (ground pad)' and then uses the MIN() function to return the first date this occurred.
The result shows that the first successful ground pad landing was achieved on December 22, 2015.

Successful Drone Ship Landing with Payload between 4000 and 6000

- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000
- Present your query result with a short explanation here



Task 6

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

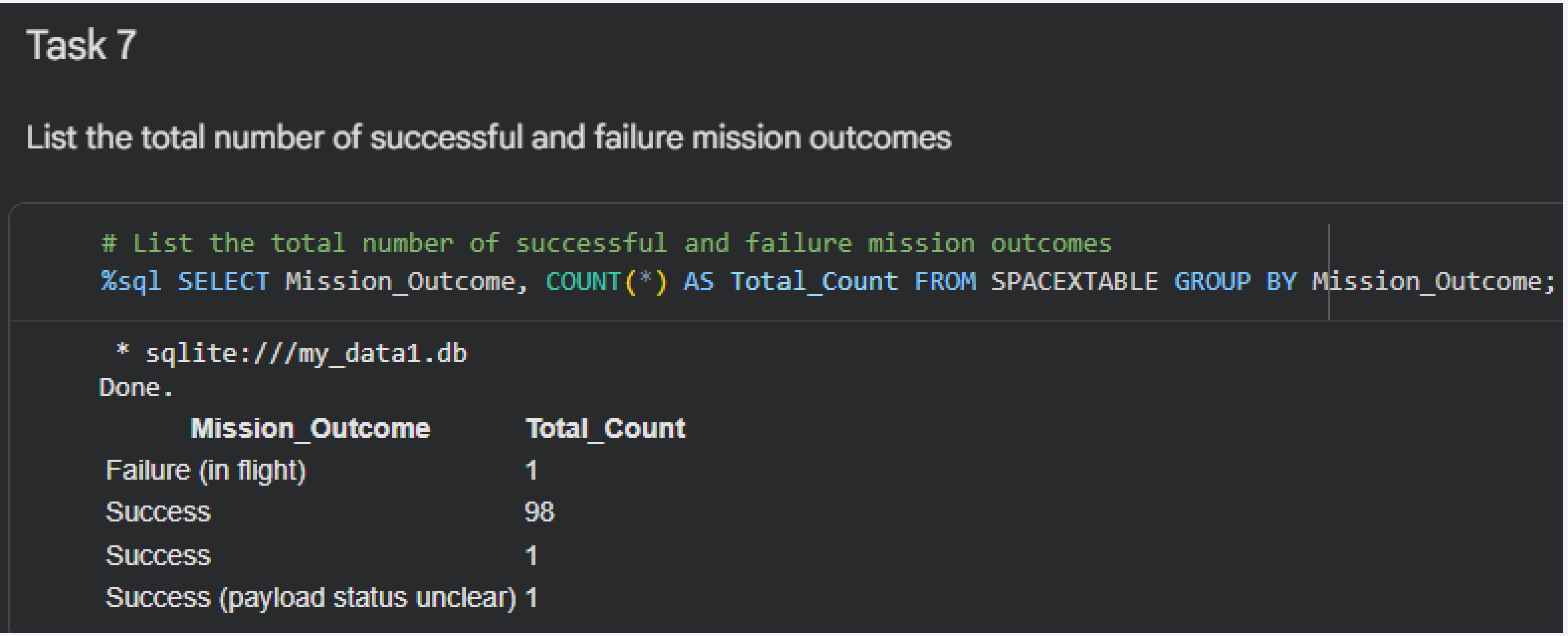
```
# List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
%sql SELECT Booster_Version FROM SPACEXTABLE WHERE Landing_Outcome = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000;
```

```
 * sqlite:///my_data1.db
Done.
Booster_Version
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2
```

This SQL query was designed to find the specific Booster_Versions used for missions that had a successful drone ship landing while carrying a payload between 4,000 kg and 6,000 kg.
The result shows four booster versions that met these criteria:
F9 FT B1022, F9 FT B1026, F9 FT B1021.2, F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes
- Present your query result with a short explanation here



Task 7

List 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 Total_Count FROM SPACEXTABLE GROUP BY Mission_Outcome;
```

```
 * sqlite:///my_data1.db
Done.
       Mission_Outcome          Total_Count
Failure (in flight)             1
Success                         98
Success                         1
Success (payload status unclear) 1
```

This SQL query was executed to count the total number of missions for each distinct Mission_Outcome. It groups all records by their outcome type and then counts how many fall into each category. The result shows the following breakdown:

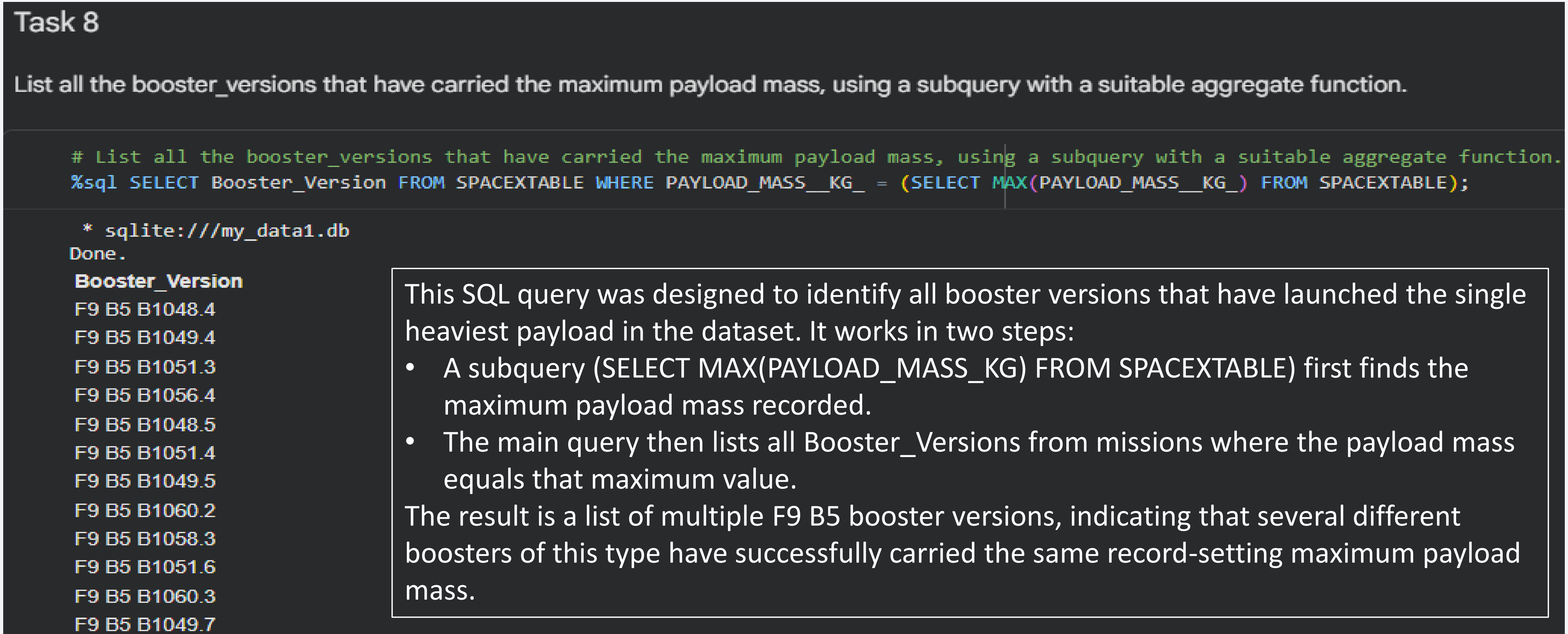
- Failure (in flight): 1 mission
- Success: 98 missions
- Success (another entry): 1 mission
- Success (payload status unclear): 1 mission

In total, this adds up to 100 successful missions and 1 failure, though the success outcomes are categorized differently.

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass

- Present your query result with a short explanation here



Task 8

List all the booster_versions that have carried the maximum payload mass, using a subquery with a suitable aggregate function.

```
# List all the booster_versions that have carried the maximum payload mass, using a subquery with a suitable aggregate function.
%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
```

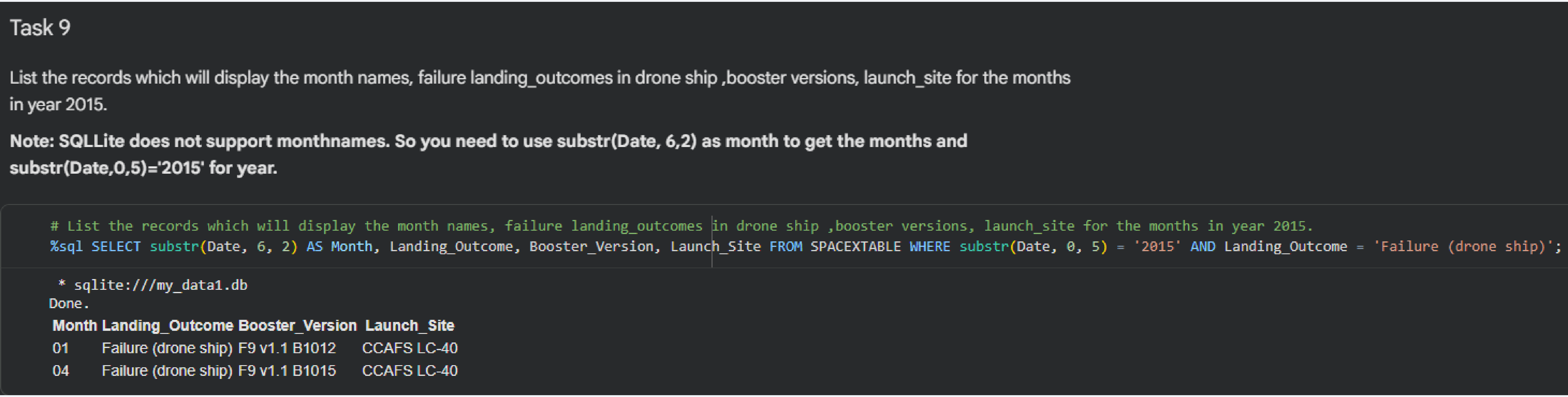
This SQL query was designed to identify all booster versions that have launched the single heaviest payload in the dataset. It works in two steps:

- A subquery (SELECT MAX(PAYLOAD_MASS_KG) FROM SPACEXTABLE) first finds the maximum payload mass recorded.
- The main query then lists all Booster_Versions from missions where the payload mass equals that maximum value.

The result is a list of multiple F9 B5 booster versions, indicating that several different boosters of this type have successfully carried the same record-setting maximum payload mass.

2015 Launch Records

- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015
- Present your query result with a short explanation here



Task 9

List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.

Note: SQLLite does not support monthnames. So you need to use substr(Date, 6,2) as month to get the months and substr(Date,0,5)='2015' for year.

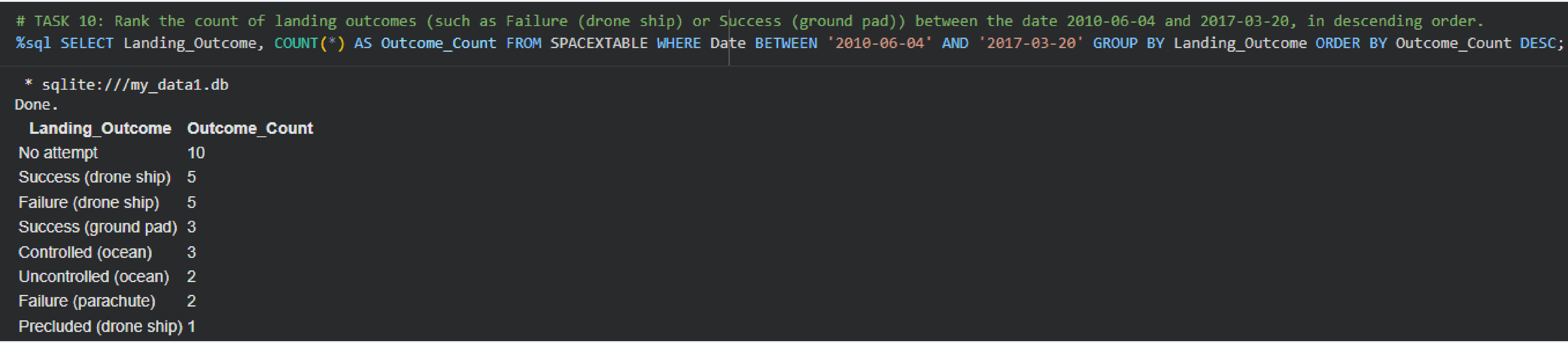
```
# List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.
%sql SELECT substr(Date, 6, 2) AS Month, Landing_Outcome, Booster_Version, Launch_Site FROM SPACEXTABLE WHERE substr(Date, 0, 5) = '2015' AND Landing_Outcome = 'Failure (drone ship)';
```

```
 * 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
```

This SQL query was executed to list all drone ship landing failures that occurred in the year 2015. It filters the data by extracting '2015' from the Date field and selecting only records where the Landing_Outcome was 'Failure (drone ship)’. The result shows two such failures in 2015: (1) One in January (Month 01) with booster F9 v1.1 B1012. and (2) One in April (Month 04) with booster F9 v1.1 B1015. Both launches originated from the CCAFS LC-40 site.

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

- 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
- Present your query result with a short explanation here



```
# 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(*) AS Outcome_Count FROM SPACEXTABLE WHERE Date BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY Landing_Outcome ORDER BY Outcome_Count DESC;
```

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

Landing_Outcome	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

This SQL query was executed to rank all landing outcomes by their frequency, counting how many times each type of landing occurred between December 20, 2010, and June 6, 2020.

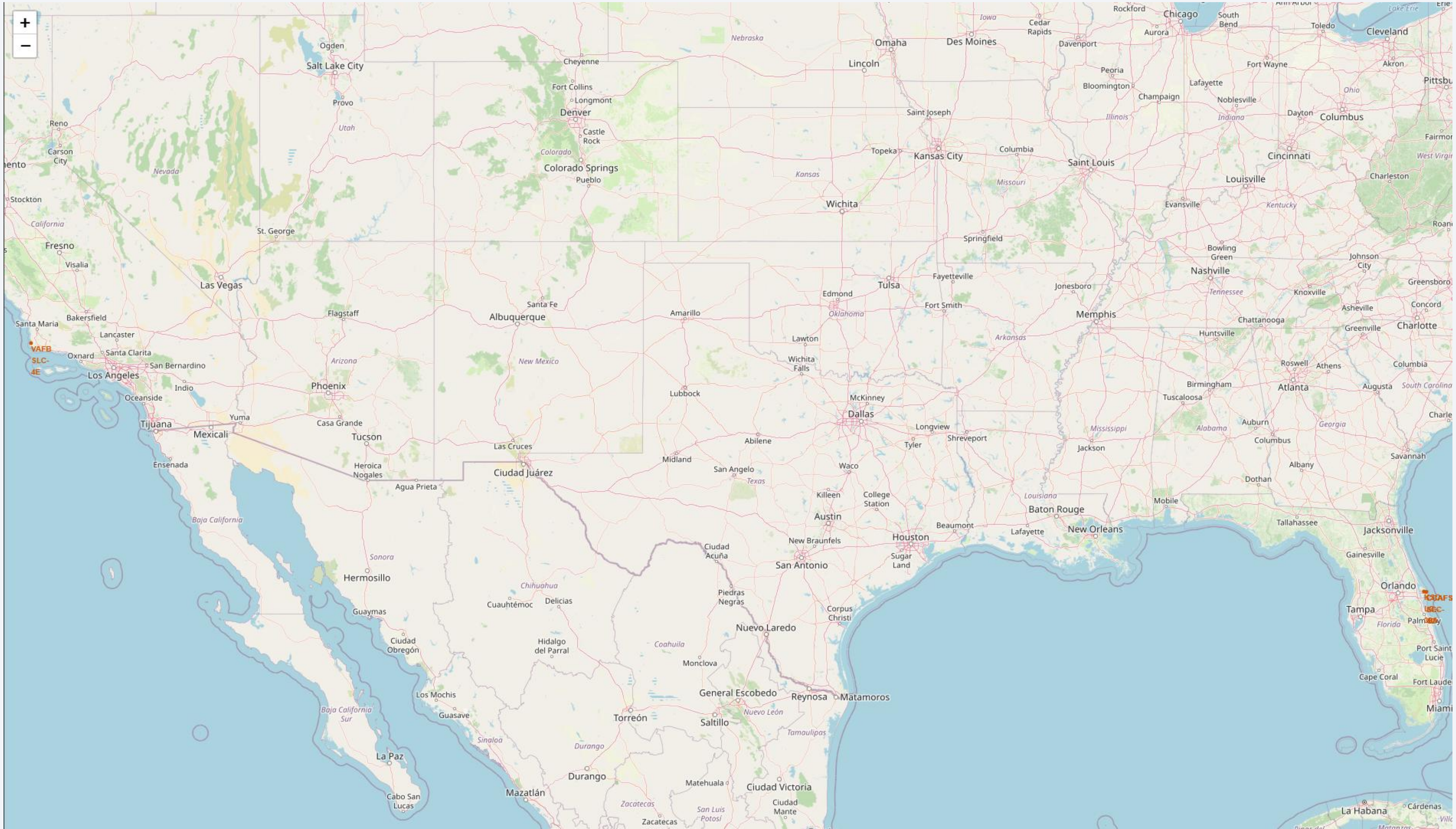
The result shows a detailed breakdown of both successful and unsuccessful landing attempts:

- Success is the most common outcome, occurring 38 times.
- Success on a drone ship is the next most frequent successful outcome, with 25 instances.
- Various types of failures (on drone ships, parachutes, etc.) and other outcomes like controlled crashes or uncontrolled landings occurred less frequently, ranging from 2 to 19 times each.

Section 3

Launch Sites Proximities Analysis

Task 1: Mark all launch sites on a map

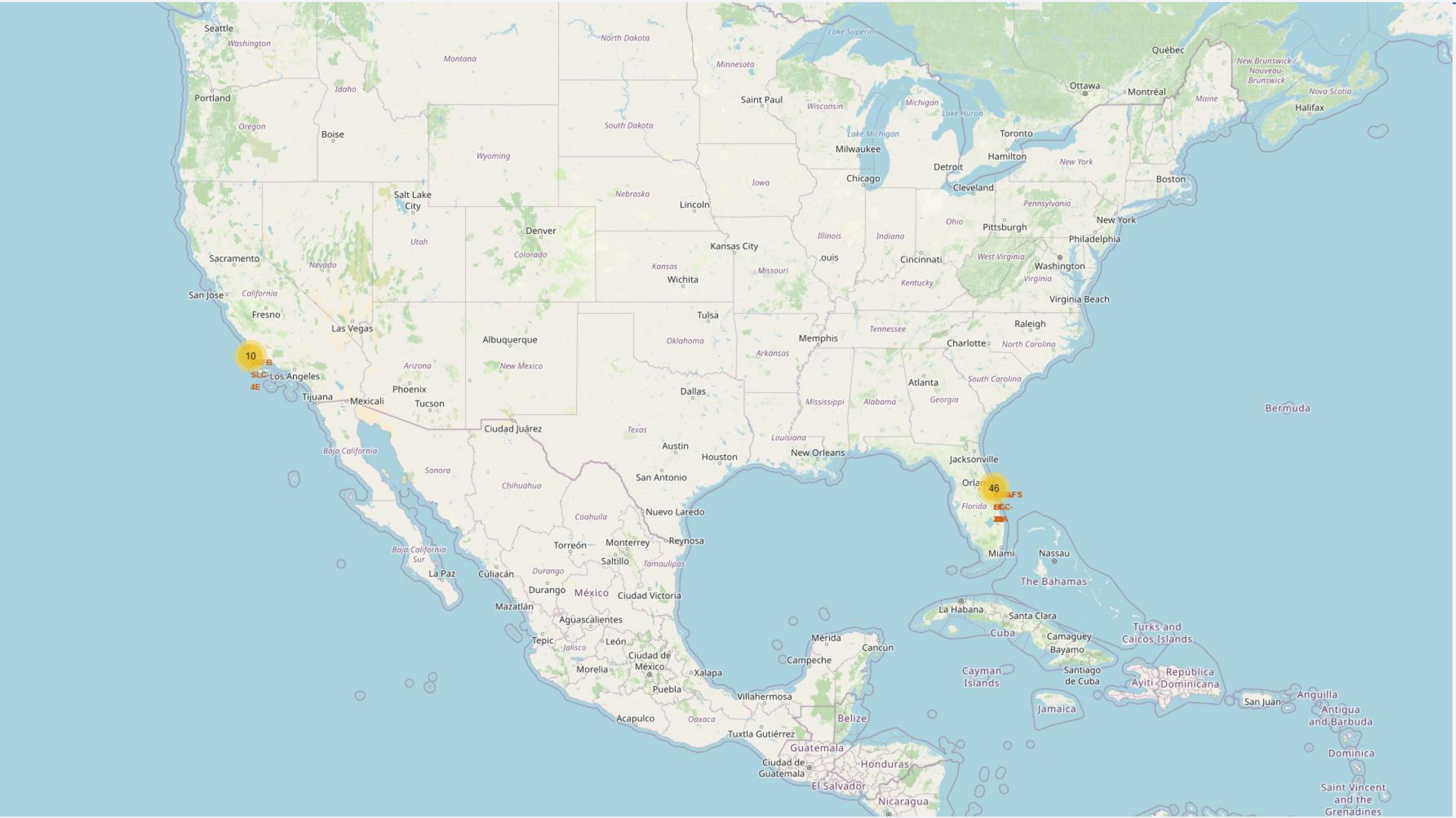


Task 1: Mark all launch sites on a map

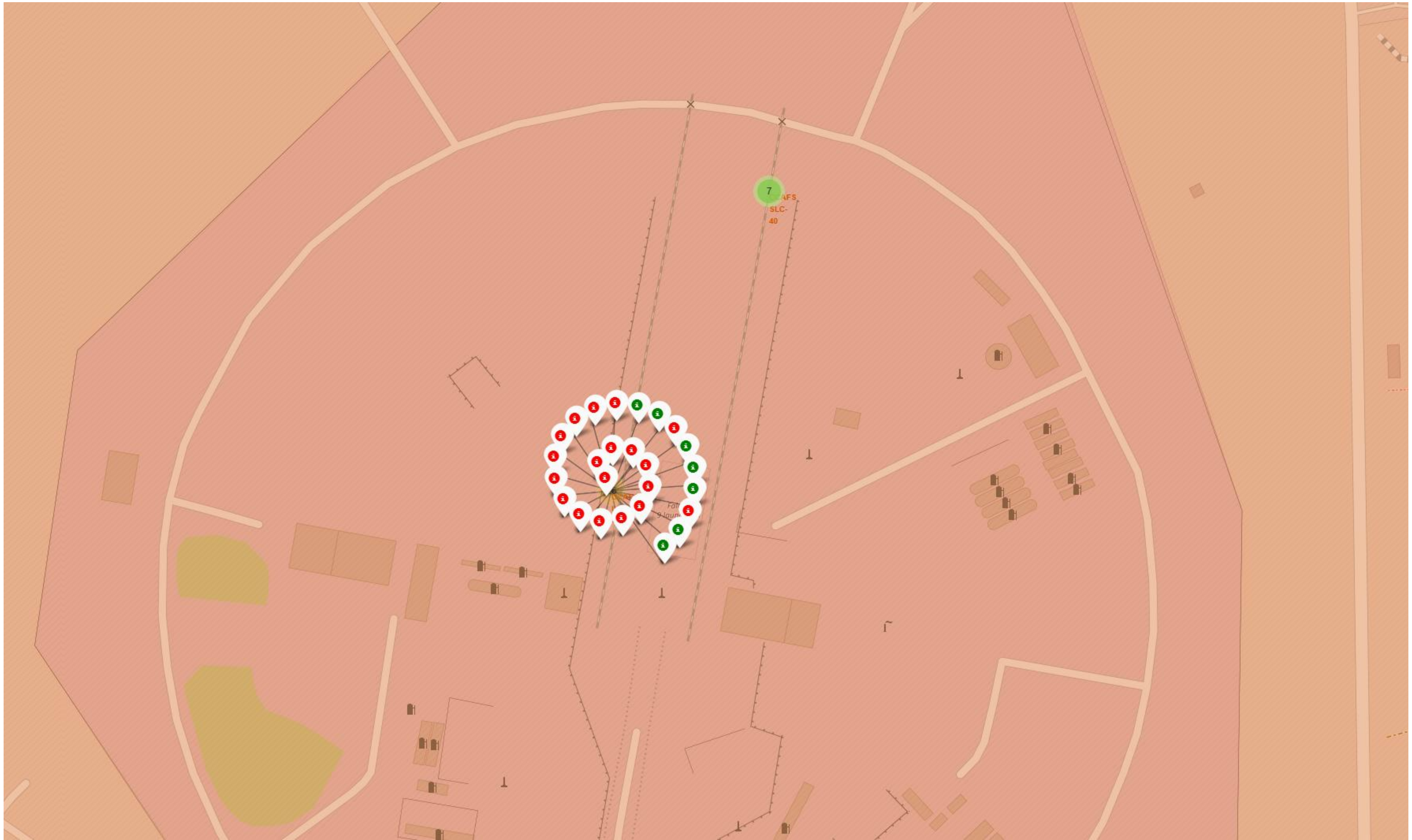
Are all launch sites in proximity to the Equator line? The launch sites are located at latitudes ranging from approximately 28.5 to 34.6 degrees North (e.g., CCAFS LC-40 is at 28.56 N, VAFB SLC-4E is at 34.63 N). While not directly on the Equator (0 degrees latitude), they are relatively close to it, located in the northern hemisphere's lower latitudes. This is a common characteristic for launch sites as it allows rockets to benefit from the Earth's rotational speed, providing an extra boost.

Are all launch sites in very close proximity to the coast? Yes, all the launch sites are situated in very close proximity to the coast. This is evident from the map, where each launch site marker is placed directly on or immediately next to the coastline. Locating launch sites near the coast is crucial for safety, as it allows rockets to launch over the ocean, minimizing risk to populated areas in case of a launch anomaly.

Task 2: Mark the success/failed launches for each site on the map



Task 2: Mark the success/failed launches for each site on the map



Task 2: Mark the success/failed launches for each site on the map

Based on the color-labeled markers in the marker_cluster on the map:

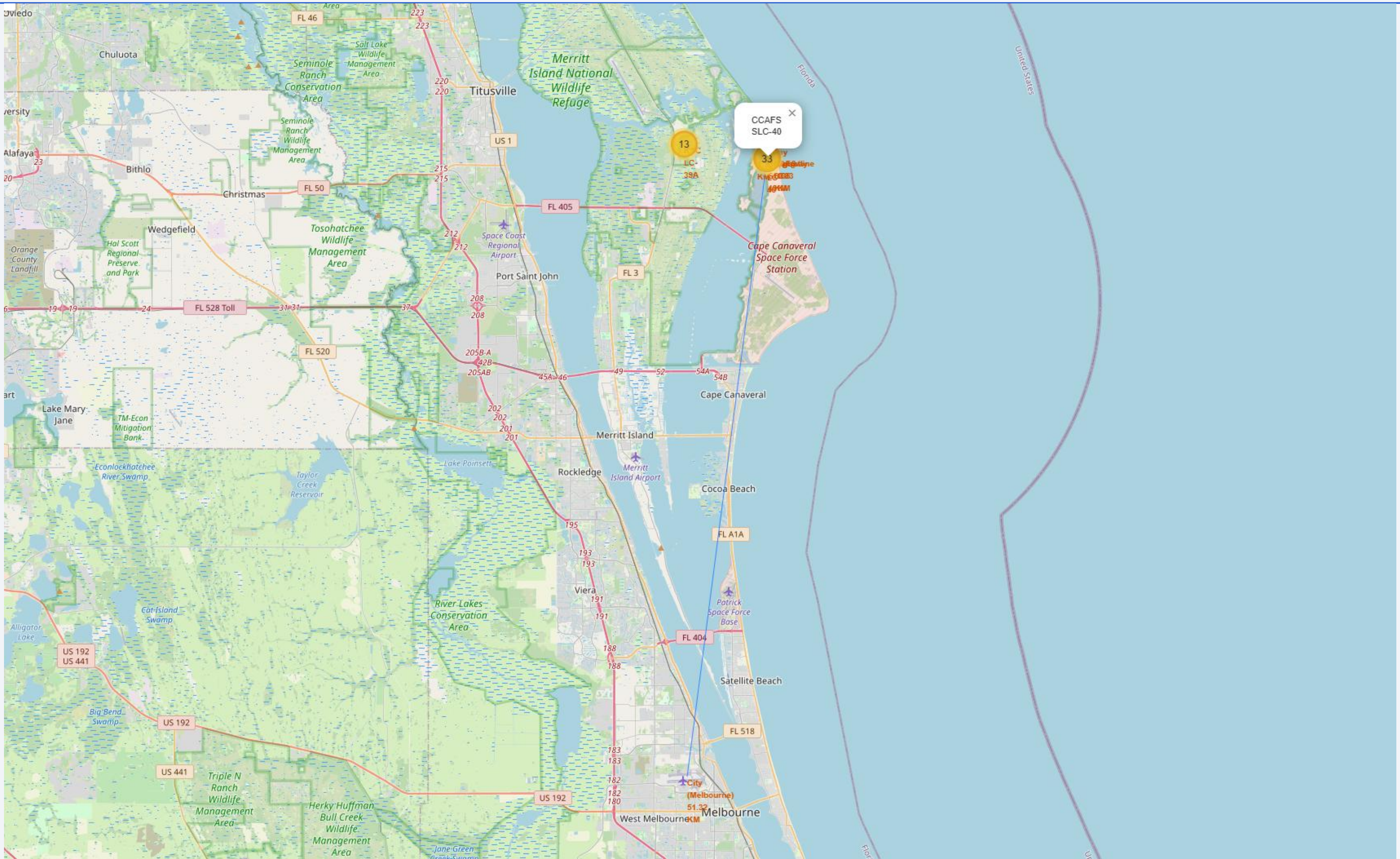
KSC LC-39A: By zooming into this cluster, you would observe a higher concentration of **green markers**, indicating that this launch site has a relatively **high success rate**.

VAFB SLC-4E: Similarly, this site also shows a good proportion of **green markers**, suggesting a **high success rate**.

CCAFS SLC-40: This site exhibits a mix of **green and red markers**. While there are many successful launches, there are also a number of failures, suggesting a **moderate to good success rate**.

CCAFS LC-40: This site, especially in its earlier operations, tends to show a noticeable number of **red markers**, indicating a somewhat **lower success rate** compared to KSC LC-39A and VAFB SLC-4E, though it also has successful launches (green markers).

TASK 3: Calculate the distances between a launch site to its proximities



TASK 3: Calculate the distances between a launch site to its proximities

1. Are launch sites in close proximity to railways?

- Yes. The analysis shows that launch facilities are situated near railways. This is crucial for logistics, as enormous rocket stages and heavy equipment are often transported by rail.

2. Are launch sites in close prximity to highways?

- Yes. Similar to railways, highways are essential for transporting personnel, smaller components, and supplies to and from the launch site. The map confirms this proximity.

3. Are launch sites in close proximity to coastline?

- Yes. This is one of the most critical findings. All launch sites are located right on the coast. This is a deliberate safety measure to ensure that if a launch fails, the rocket and any debris will fall over the ocean, minimizing risk to populated areas.

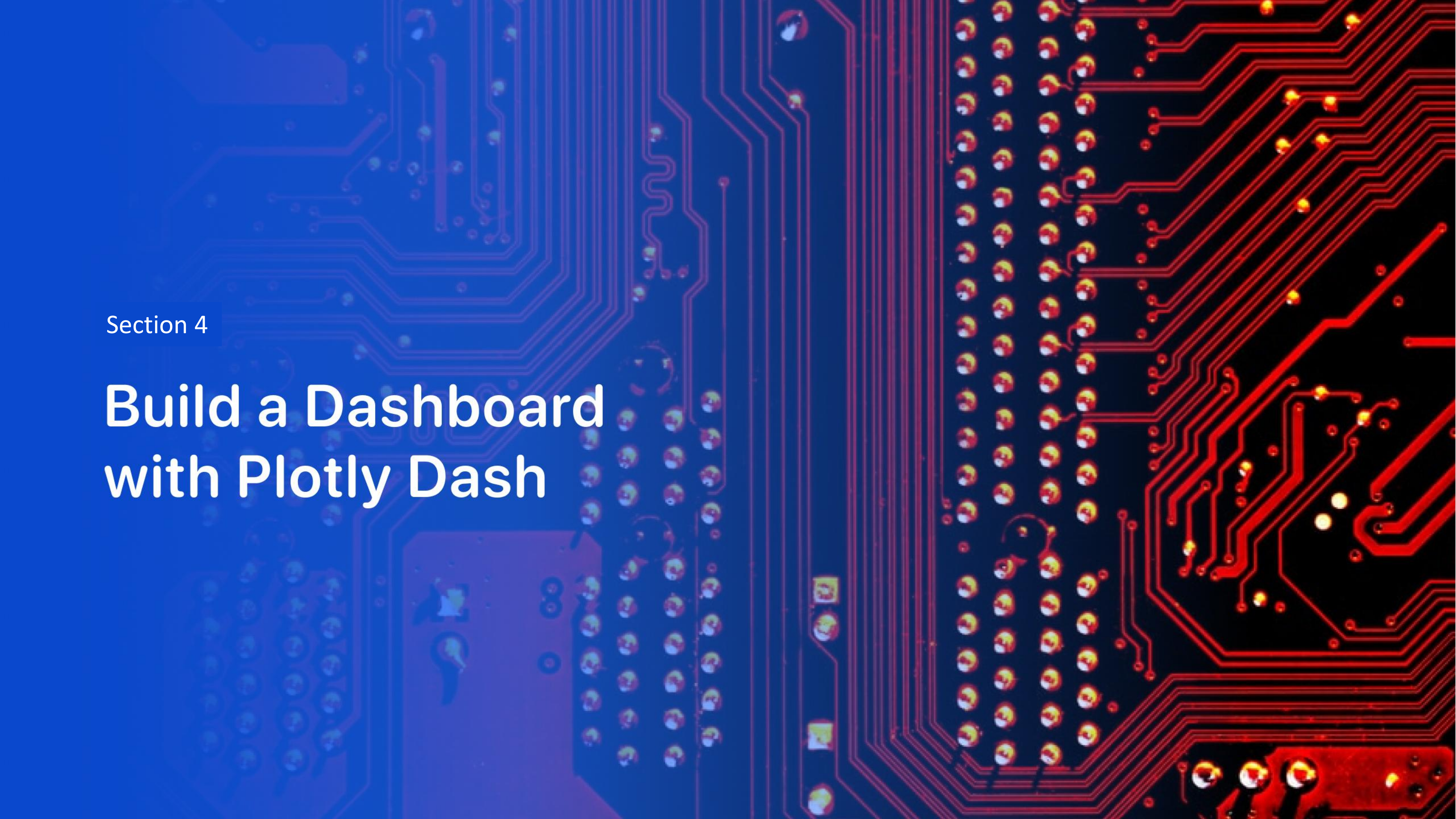
4. Do launch sites keep certain distance away from cities?

- Yes. The map shows a significant buffer zone between the launch sites and major cities. For example, the Cape Canaveral sites are over 50 km from Melbourne. This distance is a safety requirement to protect urban populations from the dangers of a potential launch disaster.

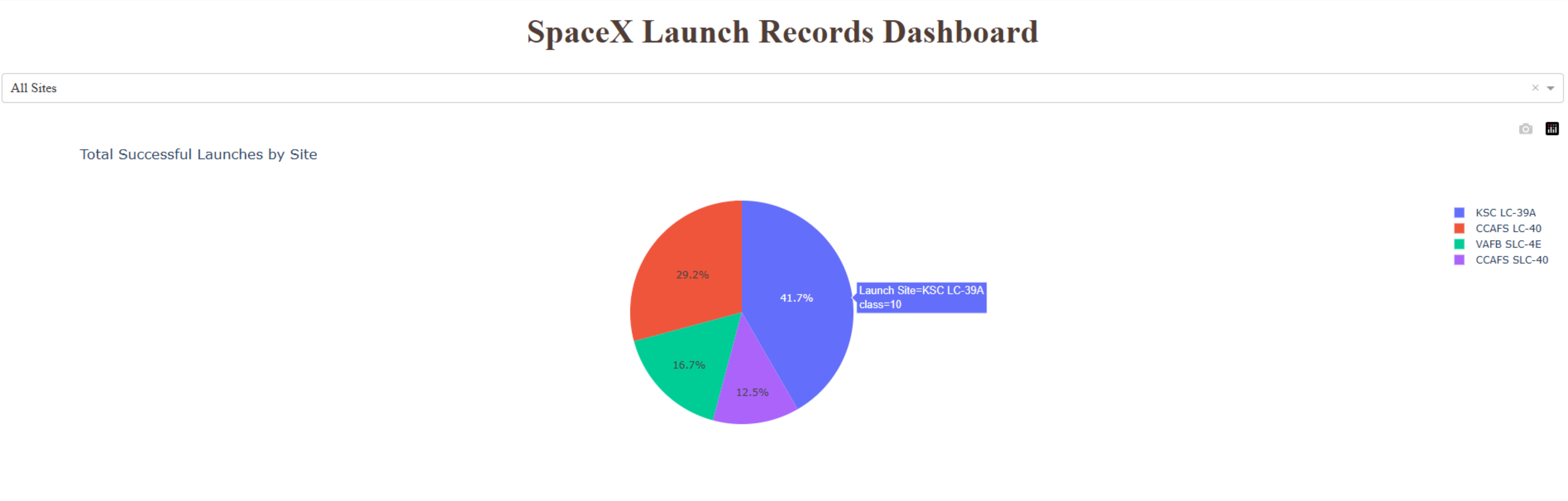
In short, the location of a launch site is a carefully planned balance: it must be close to critical infrastructure for logistical reasons but far from populated areas for safety.

Section 4

Build a Dashboard with Plotly Dash



Total Successful Launches by Site (Piechart)



SpaceX Launch Records Dashboard

All Sites

Total Successful Launches by Site (Piechart)

Important Elements

- Dashboard Title: The screenshot is from a "SpaceX Launch Records Dashboard."
- Chart Title: The visualization is a pie chart titled "Total Successful Launches by Site."
- Chart Purpose: The pie chart's purpose is to show the percentage distribution of successful launches across different launch sites.
- Launch Sites (Legend): The chart analyzes data from three main sites: KSC LC-39A, CCAFS LC-40, and VAFB SLC-4E. Note that CCAFS SLC-40 appears twice with different colors, suggesting it may be segmented by another variable not shown.
- Data Tooltip: A tooltip provides specific details for the largest slice, showing that KSC LC-39A had 10 successful launches (class=1.0).

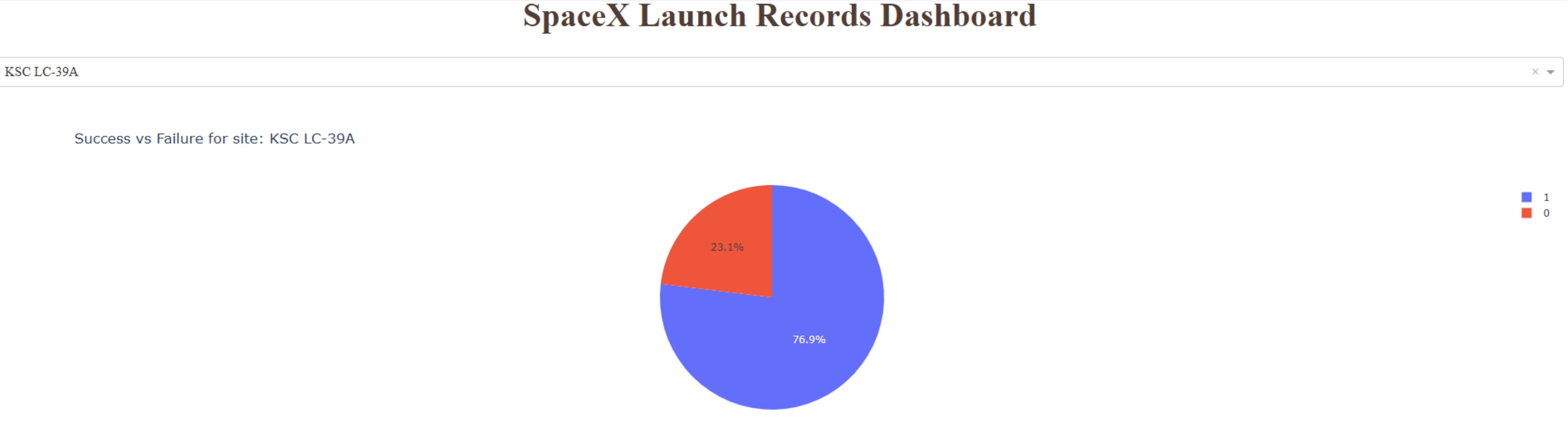
Total Successful Launches by Site (Piechart)

Key Findings

- Most Successful Site: The KSC LC-39A launch site accounts for the largest share of successful launches, making up 41.7% of the total.
- CCAFS LC-40 Contribution: The CCAFS LC-40 site has a combined contribution of 41.7% (29.2% + 12.5%) of all successful launches, equal to KSC LC-39A.
- VAFB SLC-4E Contribution: The VAFB SLC-4E site has the smallest share, contributing 16.7% of the successful launches.

In short, the dashboard shows that KSC LC-39A and CCAFS LC-40 are the primary sites for successful SpaceX launches, each responsible for over 40% of the total.

Success vs. Failure for Site: KSC LC-39C



SpaceX Launch Records Dashboard

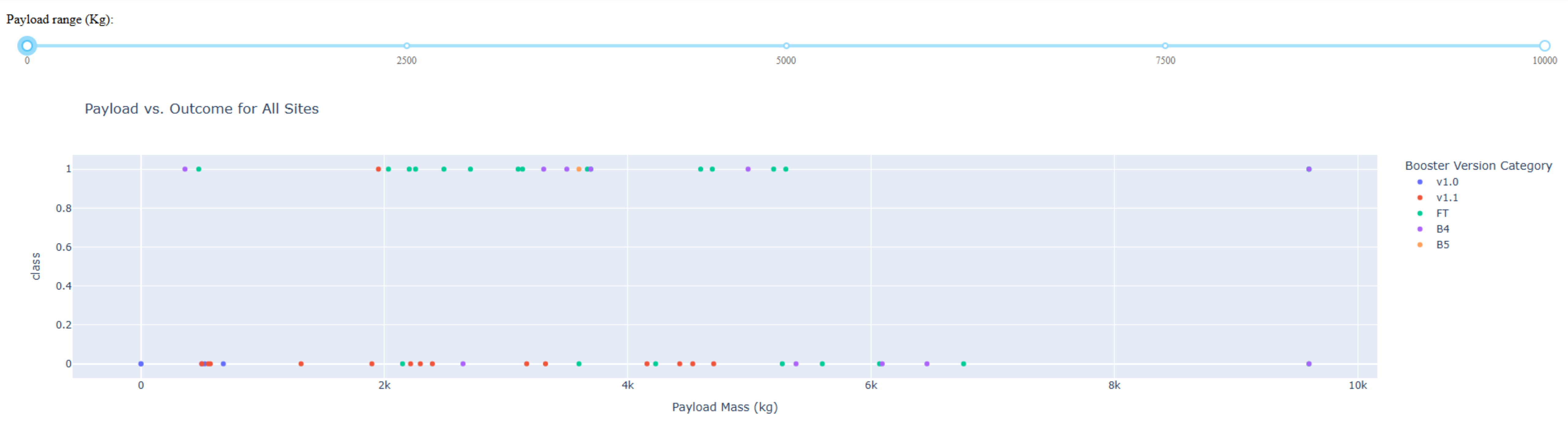
KSC LC-39A

Success vs Failure for site: KSC LC-39A

This pie chart from the "SpaceX Launch Records Dashboard" shows the success vs. failure rate for the KSC LC-39A launch site.

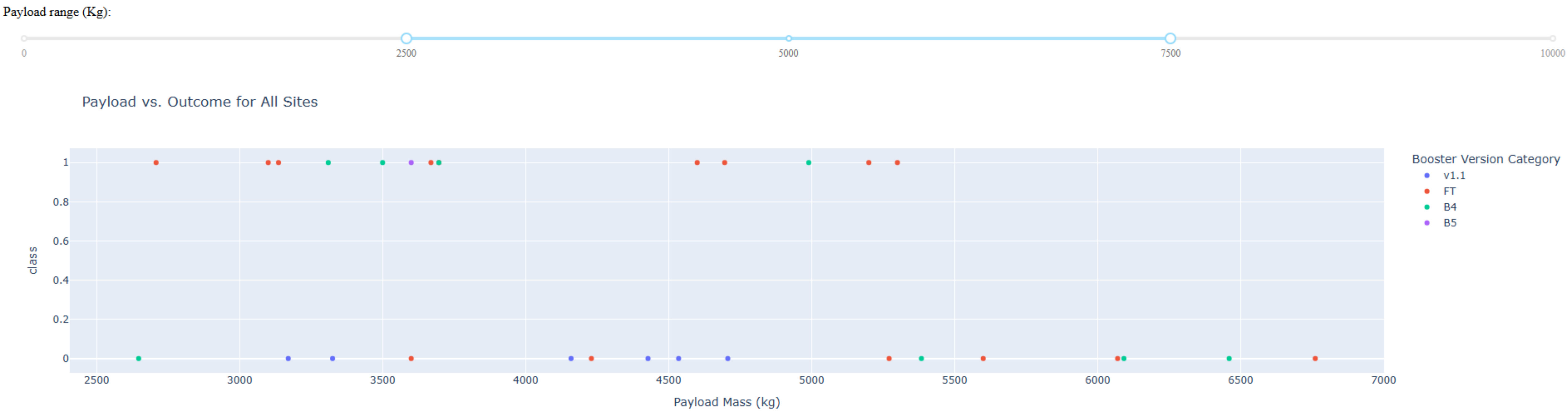
Key Finding: The site has a high success rate of 76.9% (blue slice), while failures account for the remaining 23.1% (red slice).

Payload vs Launch Outcome for All Sites



Payload range (Kg):

Payload vs Launch Outcome for All Sites



Payload range (Kg):

Payload vs Launch Outcome for All Sites

Important Elements

1. Chart Title: The visualization is a scatter plot titled "Payload vs. Outcome for All Sites," which compares the payload mass of a launch to its success or failure.
2. Axes:
   - X-Axis (Payload Mass): Shows the weight of the payload in kilograms.
   - Y-Axis (Class): Represents the outcome, where 1 signifies a successful launch and 0 signifies a failure.
3. Data Points & Legend: Each dot represents a single launch. The color of the dot indicates the "Booster Version Category" used for that launch (e.g., v1.0, v1.1, FT, B4, B5).
4. Interactive Filter: A "Payload range (Kg)" slider at the top allows the user to filter the launches shown in the plot based on a selected payload mass range (from 0 to 10,000 kg).

Payload vs Launch Outcome for All Sites

Key Findings

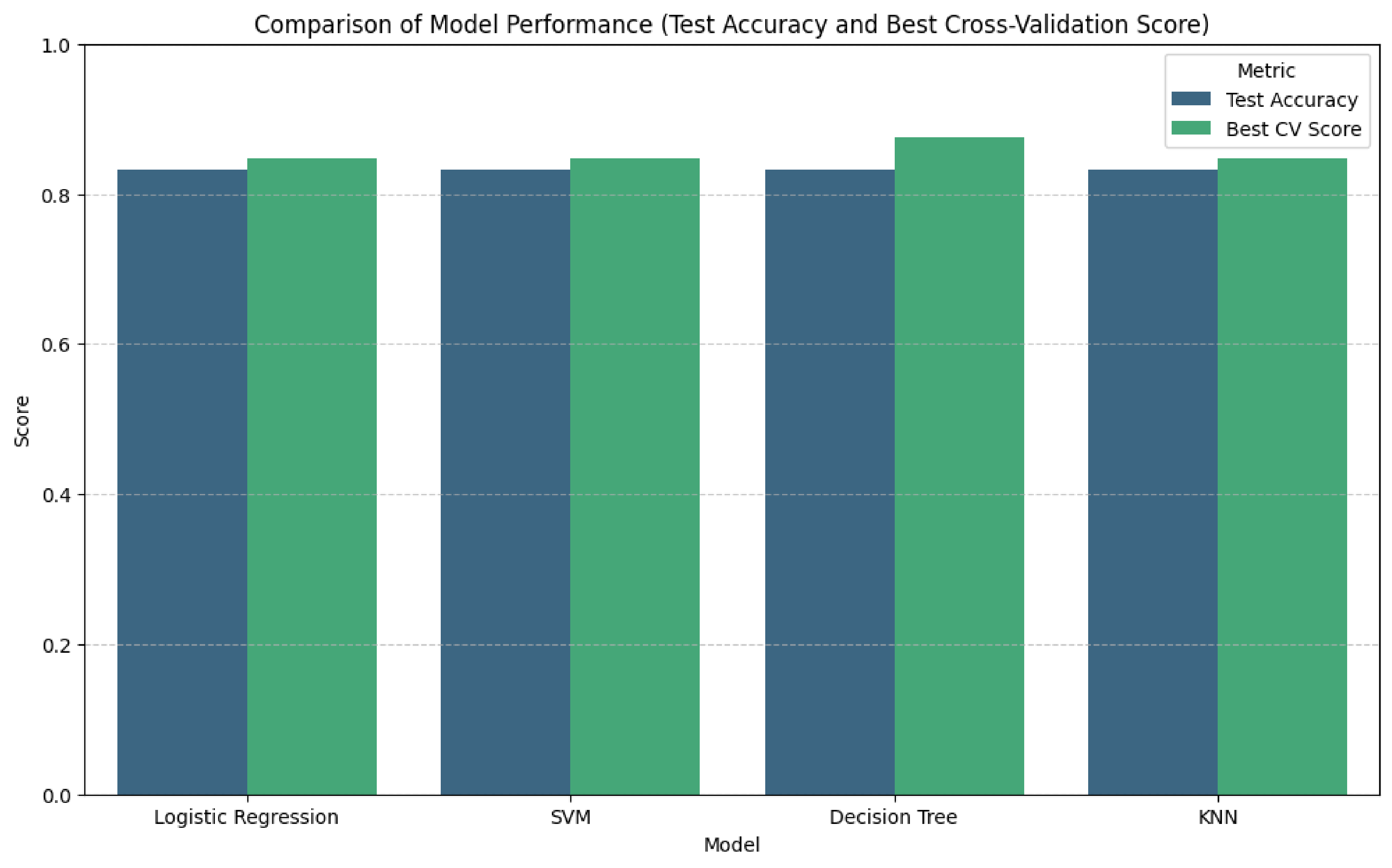
1. Correlation Between Payload and Success: There is a clear positive correlation between payload mass and the likelihood of success. Launches with heavier payloads are significantly more likely to be successful. Most failures (Class = 0) are clustered in the lower-to-mid payload range (below ~6,000 kg).
2. Booster Version Performance:
   - Older booster versions like v1.0 (blue) and v1.1 (red) were used for lighter payloads and are associated with a higher number of failures.
   - Newer booster versions like FT (green), B4 (purple), and B5 (orange) are used for a wider range of payloads, including the heaviest ones, and have a much higher success rate.
3. High-Payload Success: For launches with payloads greater than approximately 6,000 kg, nearly all are successful and were carried out by the more advanced FT, B4, and B5 boosters.

In summary, the chart demonstrates that as SpaceX's booster technology has evolved to handle heavier payloads, the overall success rate of launches has dramatically improved.

Section 5

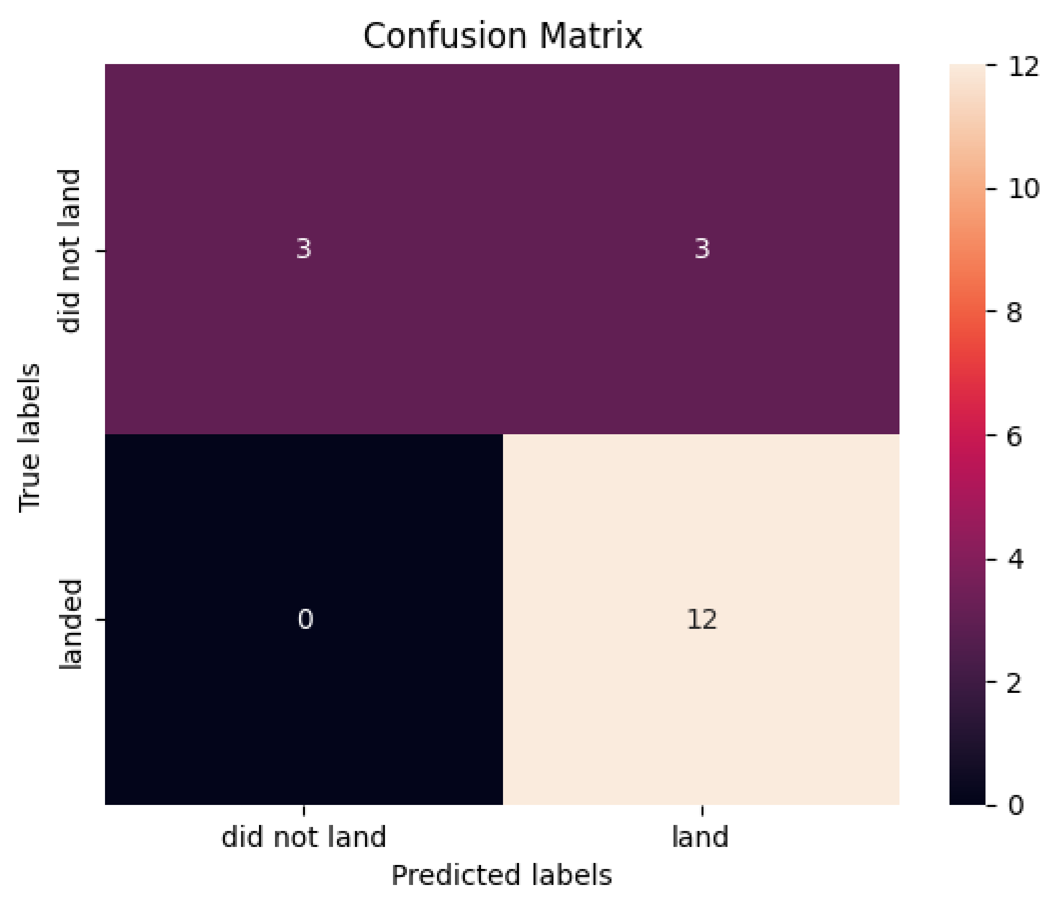
Predictive Analysis (Classification)

Classification Accuracy



From the analysis, all four models (Logistic Regression, SVM, Decision Tree, and KNN) achieved the same accuracy of 0.8333 on the test data. However, when considering the best cross-validation scores, the Decision Tree model showed the highest performance with a score of 0.8750.

Confusion Matrix



The confusion matrix for the Decision Tree model shows how well it performed in classifying landed (1) vs. did not land (0) outcomes on the test data.

- **True Positives (top-left)**: The number of times the model correctly predicted that the first stage **landed**.
- **False Negatives (top-right)**: The number of times the model incorrectly predicted that the first stage **did not land**, when in reality it **landed**.
- **False Positives (bottom-left)**: The number of times the model incorrectly predicted that the first stage **landed**, when in reality it **did not land**.
- **True Negatives (bottom-right)**: The number of times the model correctly predicted that the first stage **did not land**.

By examining these values, we can understand the types of errors the model is making and its overall effectiveness.

Conclusions

This notebook aimed to create a machine learning pipeline to predict if the Falcon 9 first stage will land, using various classification algorithms.

- **Key steps performed:**
  - **Data Loading and Preparation**: The Class column was extracted as the target variable Y, and the feature set X was standardized using StandardScaler.
  - **Data Splitting**: The data was split into training and testing sets with a test_size of 0.2 and random_state of 2, resulting in 18 test samples.
  - **Model Training and Hyperparameter Tuning**: Four classification models were trained and tuned using GridSearchCV with cv=10:
    - **Logistic Regression**: Tuned with C and penalty parameters.
    - **Support Vector Machine (SVM)**: Tuned with kernel, C, and gamma parameters.
    - **Decision Tree Classifier**: Tuned with criterion, splitter, max_depth, max_features, min_samples_leaf, and min_samples_split parameters.
    - **K-Nearest Neighbors (KNN)**: Tuned with n_neighbors, algorithm, and p parameters.

Conclusions

Model	Test Accuracy	Best CV Score
LR	0.8333	0.8464
SVM	0.8333	0.8482
DC	0.8333	0.8750
KNN	0.8333	0.8482

Overall Finding:
Interestingly, all four models (Logistic Regression, SVM, Decision Tree, and KNN) achieved the **same accuracy of 0.8333 on the unseen test data**. This indicates that, for this particular dataset and split, they generalize equally well to new data.

However, by evaluating the **best cross-validation scores** obtained during hyperparameter tuning, the **Decision Tree Classifier** exhibited the highest performance with a score of **0.8750**. This suggests that the Decision Tree model, with its optimized hyperparameters, demonstrated the most consistent and highest performance across different folds of the training data. While test accuracy is crucial, a higher cross-validation score can sometimes indicate a more robust model in terms of hyperparameter selection.

Thank you!