IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Walter R. T. Fanka
12-Dec-2024

Outline

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

Executive Summary

Summary of methodology

- This project intends to compete with SpaceX as we follow the processes involved in the methodology of Data Science; data collection, data wrangling, exploratory data analysis (EDA), data visualization, model development, model evaluation.
- It show-cases a report of results to various stakeholders in a concise and easy-to-understand approach.

Summary of all results

- The results we got through exploratory data analysis, launch site proximities, dashboard creation, and predictive analysis demonstrate our comprehensive approach to analyzing SpaceX's launch data, combining descriptive and predictive analytics to uncover valuable insights and enhance decision-making capabilities.
- The interactive visualizations and predictive models provide a robust foundation for ongoing analysis and strategic planning.

Introduction

Project background and context

- SpaceX advertises Falcon 9 rocket launches on its website with a relatively low cost ($62M unlike others with a cost upwards of $165M)
- This saving is largely in part because SpaceX can reuse the first stage.
- So, if we can determine if the first stage will land, we can determine the cost of a launch. Spaces X's Falcon 9 launch like regular rockets

Problems to find answers

- The task is to predict if the first stage of the SpaceX Falcon 9 rocket will land successfully.
- Determine the price of each launch.
- Determine if SpaceX will reuse the first stage by training a machine model using public information

Section 1

Methodology

Methodology

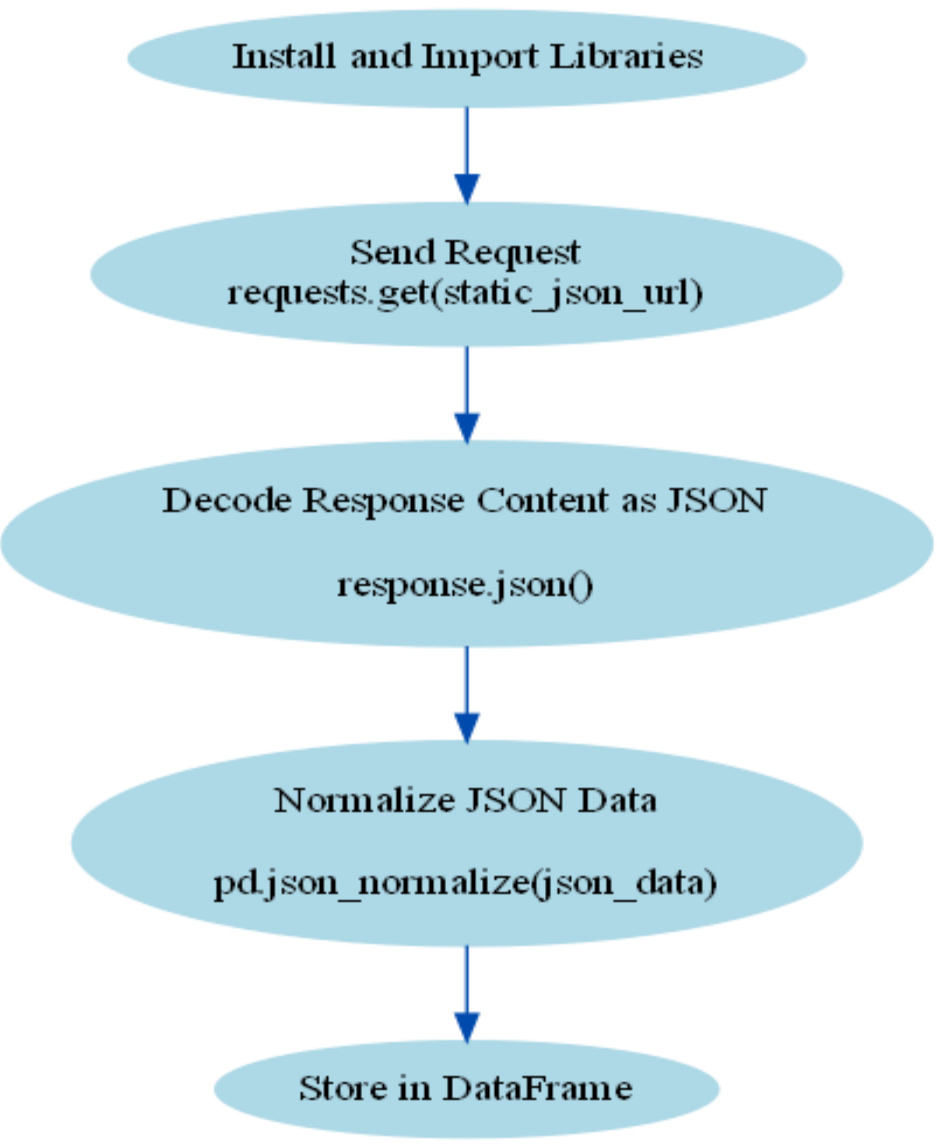
Executive Summary

The following steps will be followed in this section:

- Data collection methodology
- Perform data wrangling
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models

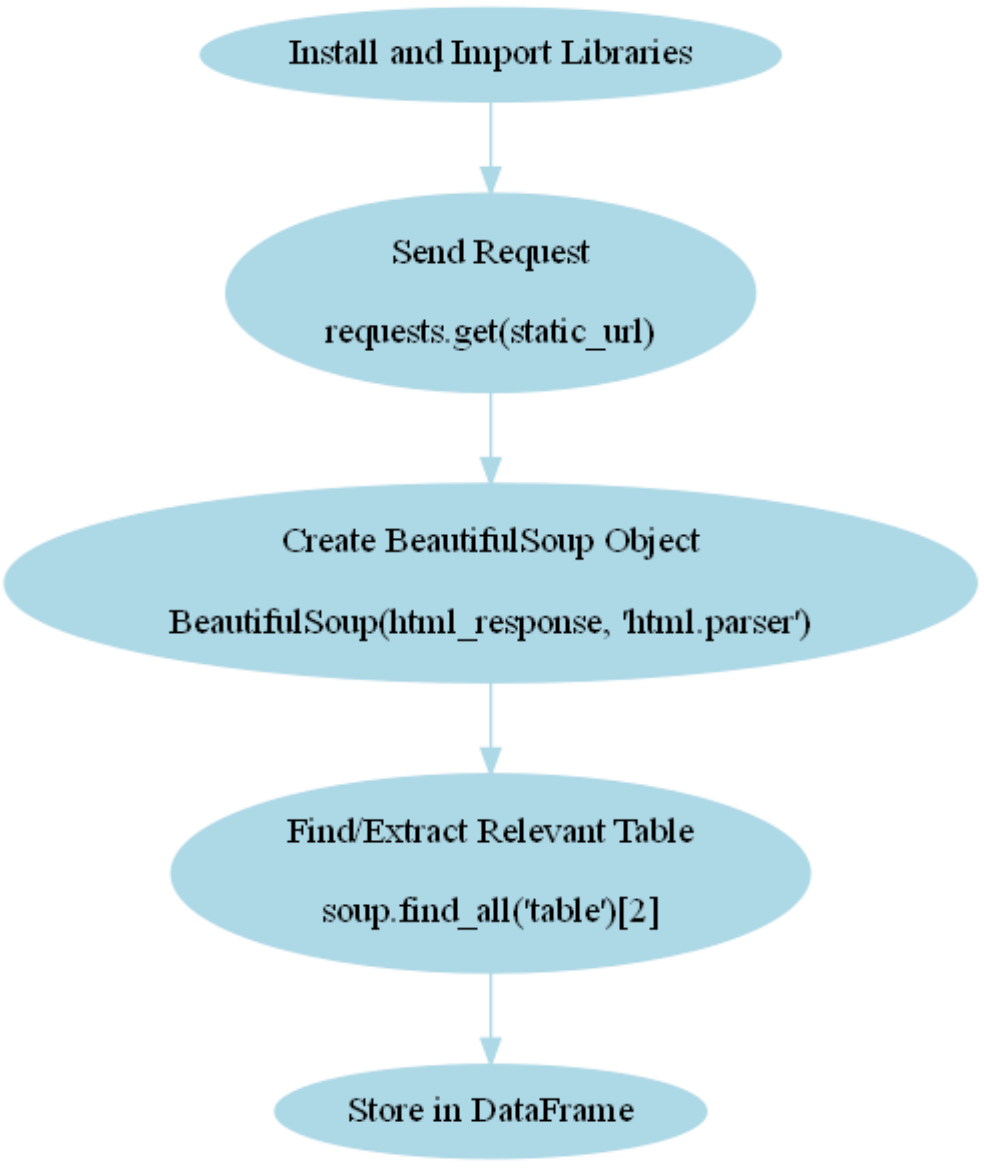
Data Collection - SpaceX API

- After installing and importing required packages, we assigned the SpaceX API to the variable `static_json_url.`
- Using the `requests.get()` method, we made a request to the url to get a response.
- Then we decode the `response` content as a Json using `.json()` method. This is the json data.
- Finally we turn the json data into a Pandas data frame using the `.json_normalize()` method.
- Click here to navigate to the GitHub URL of the completed web scraping notebook.



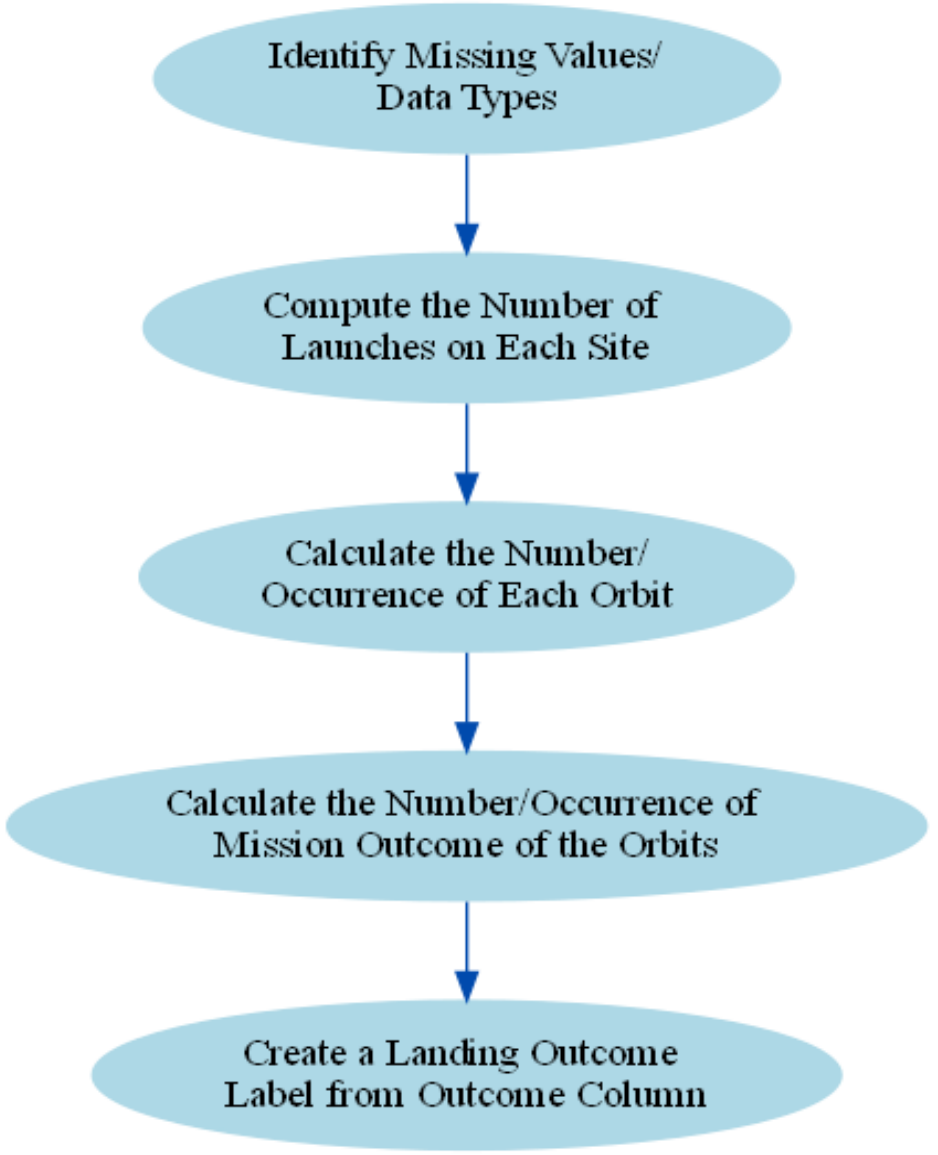
Data Collection - Scraping

- After installing and importing required packages, we assigned the Falcon9 launch HTML page to the variable `static_url`.
- Using the `requests.get()` method, we made a request to the url to get a HTML response.
- Now we build a BeautifulSoup object from the response content.
- Then we find the desired table and extract all column names from the HTML table header.
- Finally, we create a data frame by parsing the launch HTML tables
- Click here to navigate to the GitHub URL of the completed web scraping notebook.



Data Wrangling

- We calculated the percentage of missing values using .isnull(), .sum(), and len() methods:
  `df.isnull().sum()/len(df)100`
- Next we calculated the number of launches on each site:
  `df.LaunchSite.value_counts()`
- Then we compute the number and occurrence of mission outcome of the orbits:
  `df.Orbit.value_counts()`
- Finally, we create a landing outcome from the Outcome column, which takes the values of either 1 or 0.
- Click here to navigate to the GitHub URL of the completed data wrangling notebook.



EDA with Data Visualization

Summary of Plotted Charts

1. Relationship between Flight Number and Launch Site:
    - Scatter Plot to visualize the distribution of flight numbers across different launch sites and identify any patterns or clustering in the data.
2. Relationship between Payload and Launch Site:
    - Bar Chart to compare the payload capacities for different launch sites, providing insights into which sites are preferred for heavier or lighter payloads.
3. Relationship between Success Rate of Each Orbit Type:
    - Pie Chart to depict the success rates of various orbit types, helping to determine which orbits have higher success rates and thus might be more reliable.
4. Relationship between Flight Number and Orbit Type:
    - Line Chart to examine how the flight numbers are distributed among different orbit types, potentially revealing trends or preferences for certain orbits over time.

EDA with Data Visualization

Summary of Plotted Charts

5. Relationship between Payload and Orbit Type:

- Box Plot to show the distribution and range of payload weights across different orbit types, highlighting any significant differences or outliers.

6. The Launch Success Yearly Trend:

- Line Chart to track the success rates of launches over the years, identifying trends and improvements in launch success over time.

These charts collectively provide a comprehensive analysis of various aspects of the launches, from site performance to payload distribution, orbit success rates, and yearly trends, offering valuable insights for strategic decision-making and optimization. (GitHub link)

EDA with SQL

Summary of SQL Queries

1. Table Management:

- Dropped existing table and created a new table with non-null dates.

2. Launch Site Analysis:

- Retrieved distinct launch sites.
- Filtered and displayed launch sites starting with 'CCA'.

3. Payload Mass Calculations:

- Total payload mass for NASA (CRS).
- Average payload mass for Booster Version 'F9 v1.1'.

4. Landing Outcomes:

- Date of first successful ground pad landing.
- Booster versions with successful drone ship landings (payload 4000-6000 kg).

5. Mission Outcomes:

- Counted total mission outcomes.
- Identified booster version with maximum payload mass.

6. Specific Year Analysis:

- Analyzed failed drone ship landings in 2015.
- Counted landing outcomes from 2010-06-04 to 2017-03-20.

(GitHub Link)

Build an Interactive Map with Folium

Summary of Added Map Objects

1. Markers:
   - Purpose: Pinpoint specific locations of interest (e.g., launch sites, key landmarks).
   - Usage: Added to highlight and provide details about each site with a clickable popup.
2. Circles:
   - Purpose: Represent areas around a point, often to indicate zones or regions (e.g., safety zones, impact areas).
   - Usage: Added to show the radius of influence or coverage around each site, with varying radii and colors for distinction.
3. Lines/Polylines:
   - Purpose: Connect different points to indicate routes or paths (e.g., flight paths, travel routes).
   - Usage: Added to visualize connections between launch sites and target locations or to map out planned trajectories.

Build an Interactive Map with Folium

Explanation for Adding These Objects

- Markers: To provide clear, interactive points on the map that users can click to get detailed information about specific locations, such as launch sites or key points of interest.
- Circles: To illustrate the coverage area or zones around a specific point, helping users understand the spatial influence or safety boundaries related to those locations.
- Lines/Polylines: To depict routes or connections between multiple points, useful for visualizing flight paths, travel routes, or connections between different sites. This helps in understanding the relationships and distances between these points.

These objects help in creating a comprehensive and interactive map, making it easier to visualize spatial relationships, coverage areas, and specific locations with additional context and detail. (GitHub link)

Build a Dashboard with Plotly Dash

Summary of Plots/Graphs and Interactions Added to a Dashboard

1. Bar Chart:
   - Displays the total payload mass for different customers.
   - Used to compare the payload mass handled by different customers, providing insights into which customers are contributing the most to the payload volume.
2. Line Chart:
   - Shows the launch success rate over the years.
   - Used to identify trends and improvements in launch success rates over time, highlighting periods of high and low performance.
3. Pie Chart:
   - Illustrates the distribution of different mission outcomes.
   - Used to give a quick visual summary of the proportion of various mission outcomes, helping to assess overall mission success.

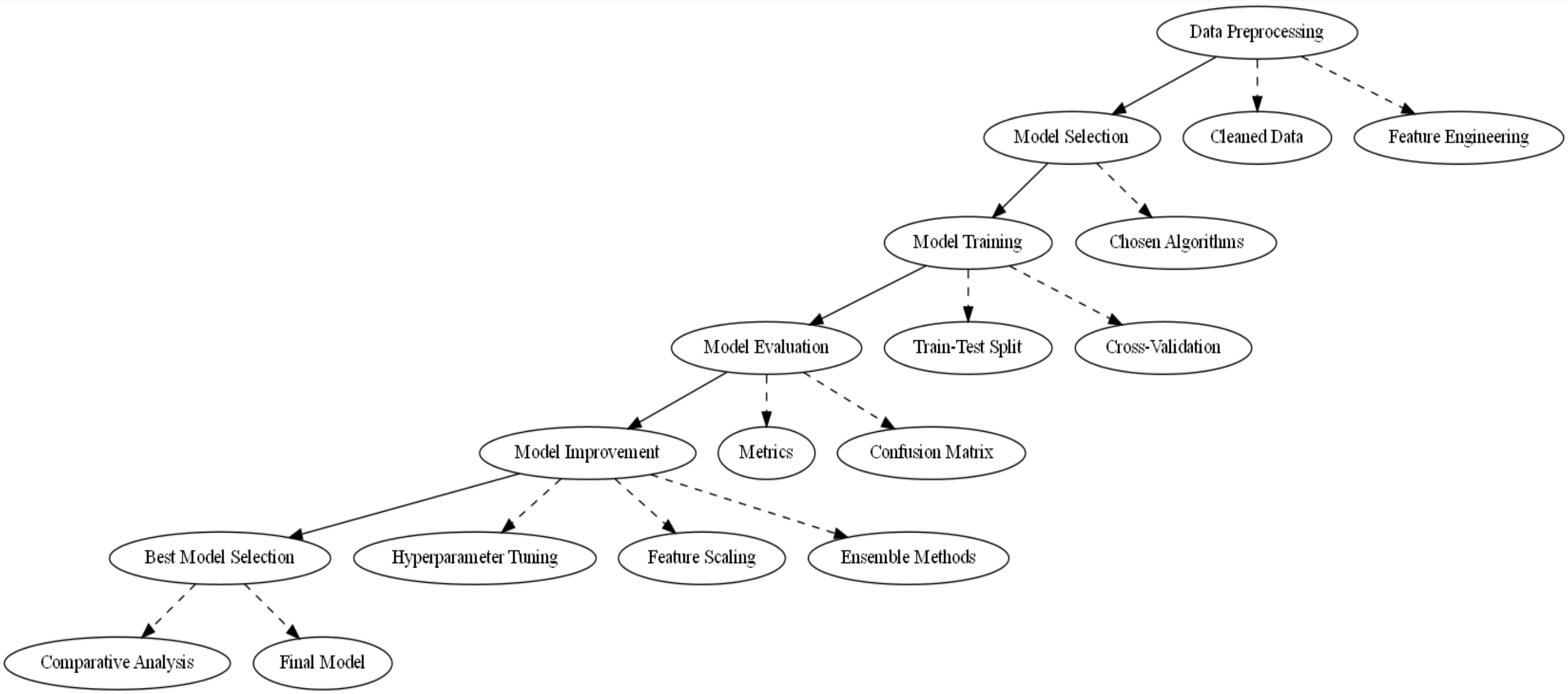
4. Scatter Plot:
   - Plots the relationship between payload mass and launch sites.
   - Used to identify any correlations between payload mass and specific launch sites, aiding in understanding site preferences for different payload weights.
5. Heat Map:
   - Displays the frequency of launches from different sites.
   - To visualize the intensity and distribution of launch activities across various sites, identifying key operational areas.
6. Interactive Filters:
   - Allows users to filter data by date range, launch site, and mission outcome.
   - To enable users to customize their view and focus on specific data points, enhancing the usability and flexibility of the dashboard. (GitHub Link)

Predictive Analysis (Classification)–Key Phases

Summary of Building, Evaluating, and Improving a Classification Model

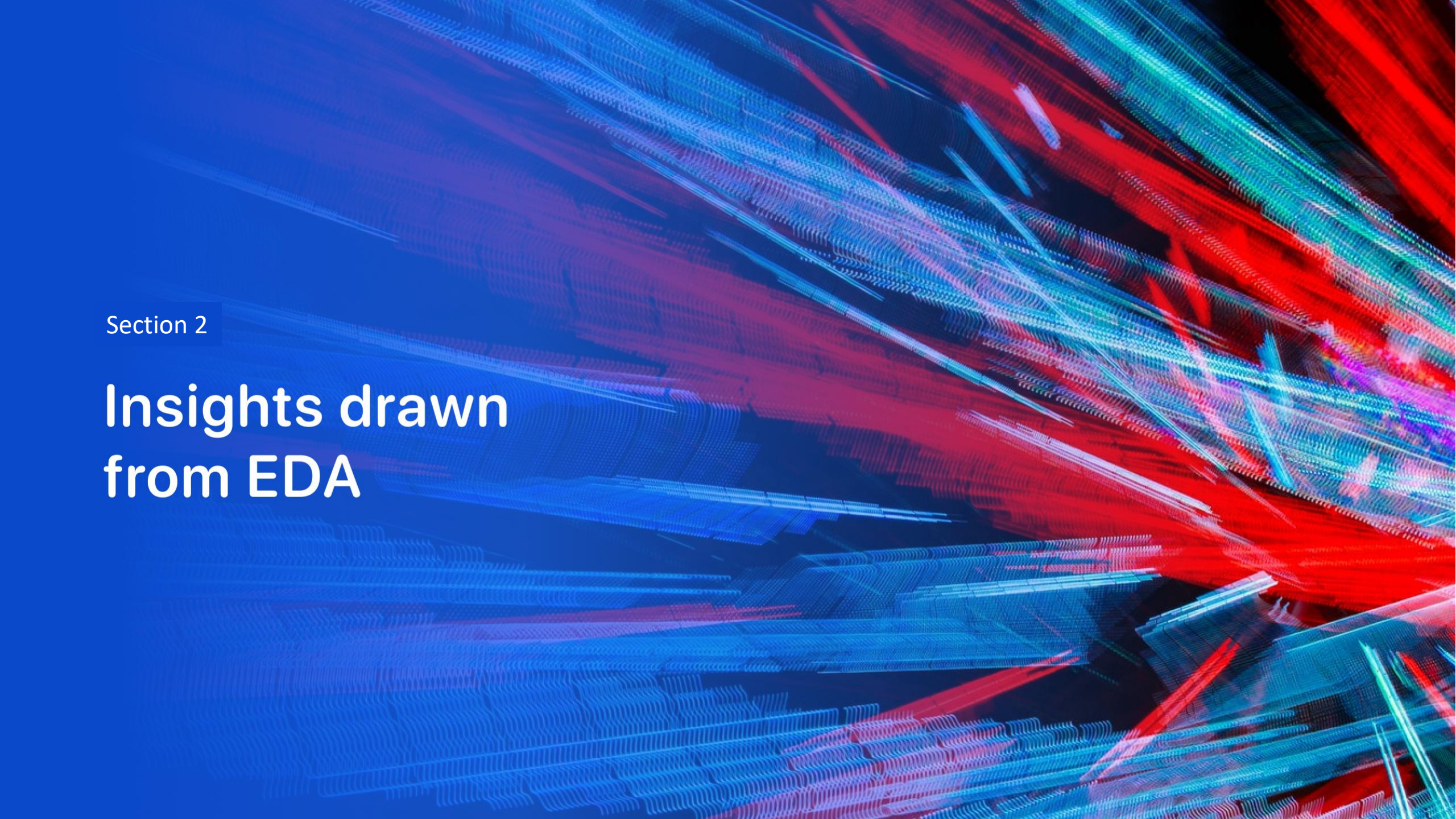
1. Data Preprocessing:
   - Cleaned Data: Handled missing values, outliers, and standardized features.
   - Feature Engineering: Created new features and selected relevant ones.
2. Model Selection:
   - Chose Algorithms: Tested different classification algorithms (e.g., Logistic Regression, Decision Tree, Random Forest).
3. Model Training:
   - Train-Test Split: Divided data into training and testing sets.
   - Cross-Validation: Used k-fold cross-validation to ensure robust performance.
4. Model Evaluation:
   - Metrics: Evaluated models using accuracy, precision, recall, and F1-score.
   - Confusion Matrix: Analyzed true positives, false positives, true negatives, and false negatives.
5. Model Improvement:
   - Hyperparameter Tuning: Used GridSearchCV.
   - Feature Scaling: Applied StandardScaler.
6. Best Model Selection:
   - Comparative Analysis: Compared models based on performance metrics.
   - Final Model: Selected the best-performing model and validated it on a hold-out set. (GitHub Link)

Predictive Analysis (Classification)–Flowchart

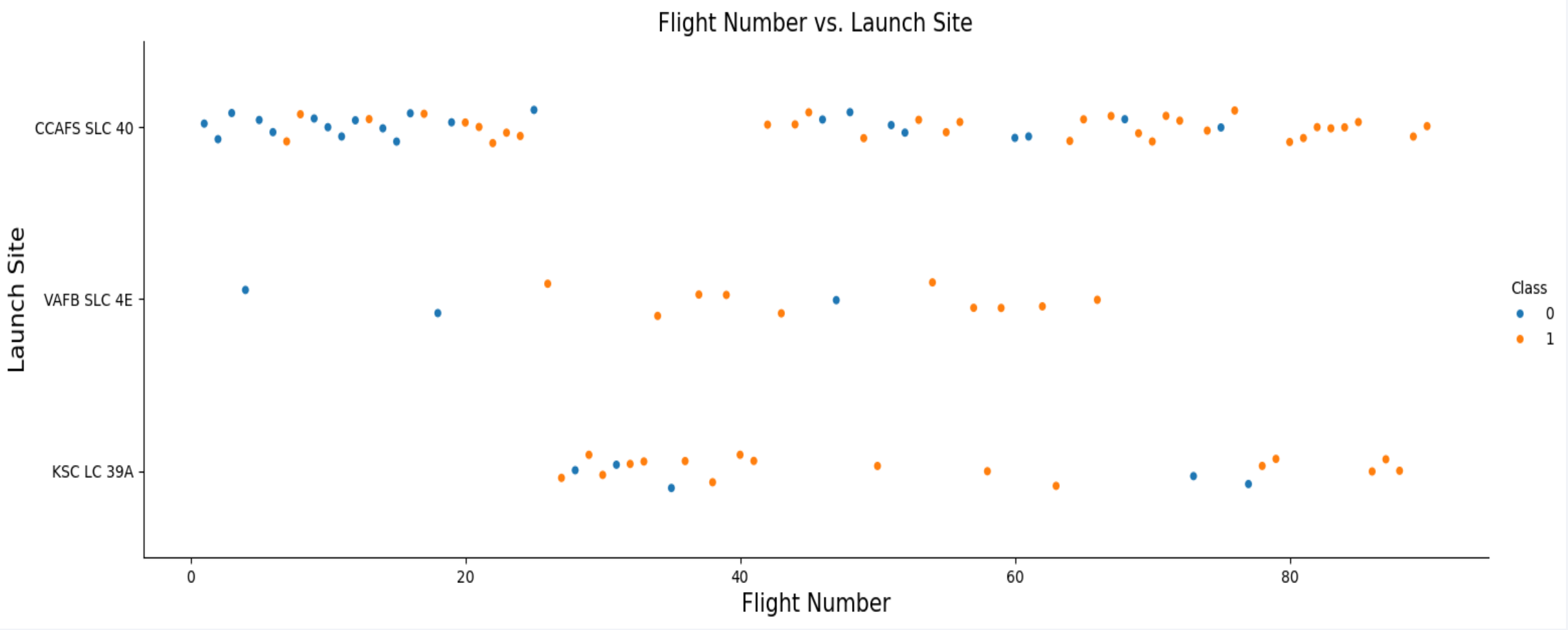


Results

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


Section 2
Insights drawn from EDA

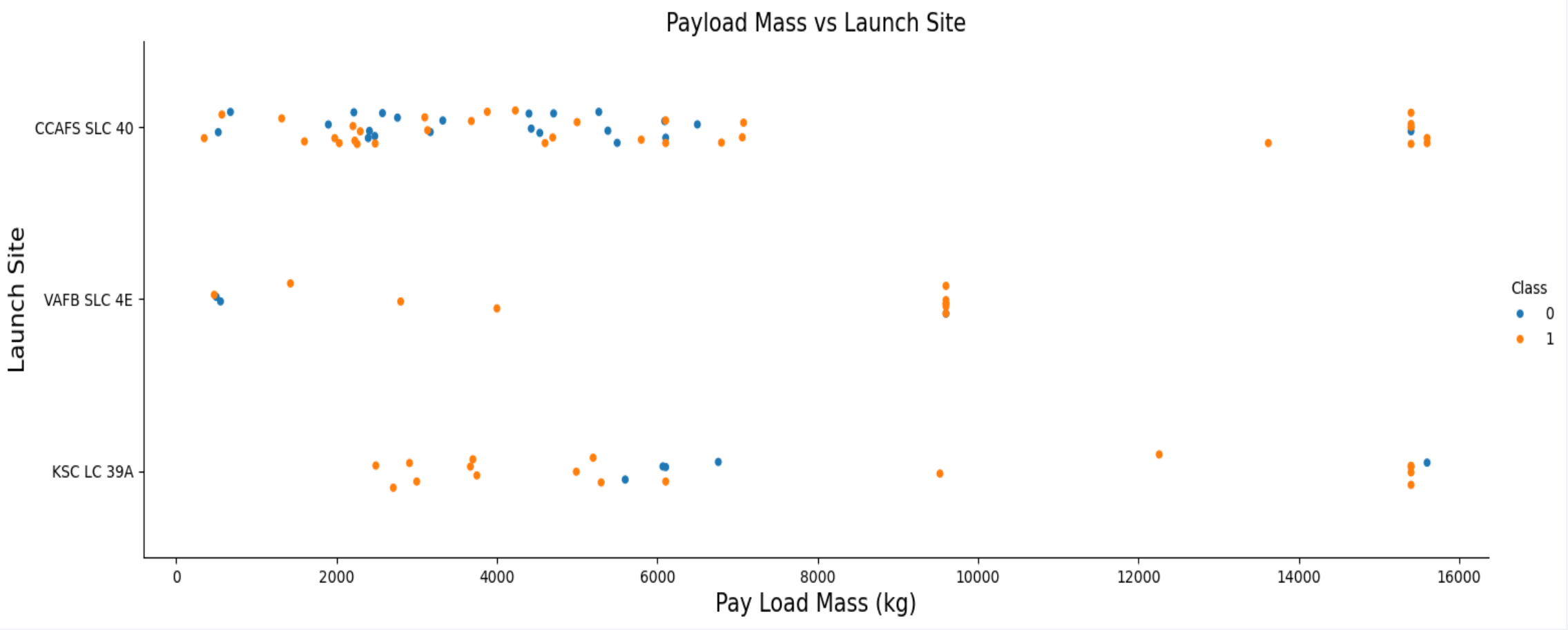
Flight Number vs. Launch Site



- X-axis represents the Flight Number and Y-axis represents the Launch Site. Each point represents a flight, positioned by flight number and launch site. Blue points indicate Class 0 (failures) while orange indicates successes.

- **Purpose:** The plot visualizes the distribution of flight numbers across different launch sites and their corresponding success or failure outcomes. Examining this plot, we see that at higher flight numbers CCAFS SLC 40 and VAFB SLC 4E achieve more successful launches.

Payload vs. Launch Site



- X-axis represents the Pay Load Mass and Y-axis represents the Launch Site. Each point represents a flight, positioned by pay load mass and launch site. Blue points indicate Class 0 (failures) while orange indicates successes.

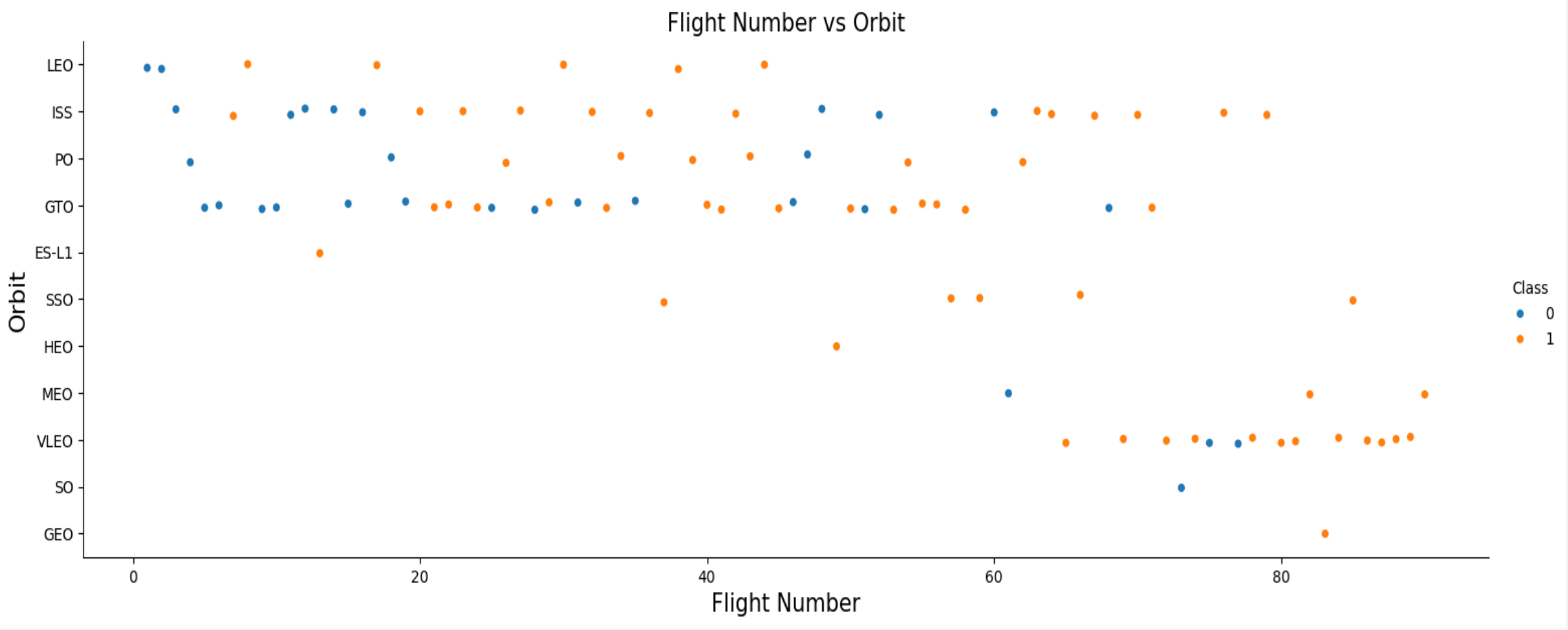
- **Purpose:** The plot visualizes the distribution of pay load masses across different launch sites and their corresponding success or failure outcomes. Examining this plot, we see that at higher pay load masses, all VAFB SLC 4E launches were successful.

Success Rate vs. Orbit Type

- X-axis represents different types of orbits. and Y-axis represents the success rate, ranging from 0 to 1. **ESL1, GEO, HEO, and SSO** have the highest success rates, close to 1.

- **Purpose:** Visualizes the success rate of missions for different orbital types. Analyzing this data helps in identifying which orbital missions have historically higher success rates and which ones may require further improvements.

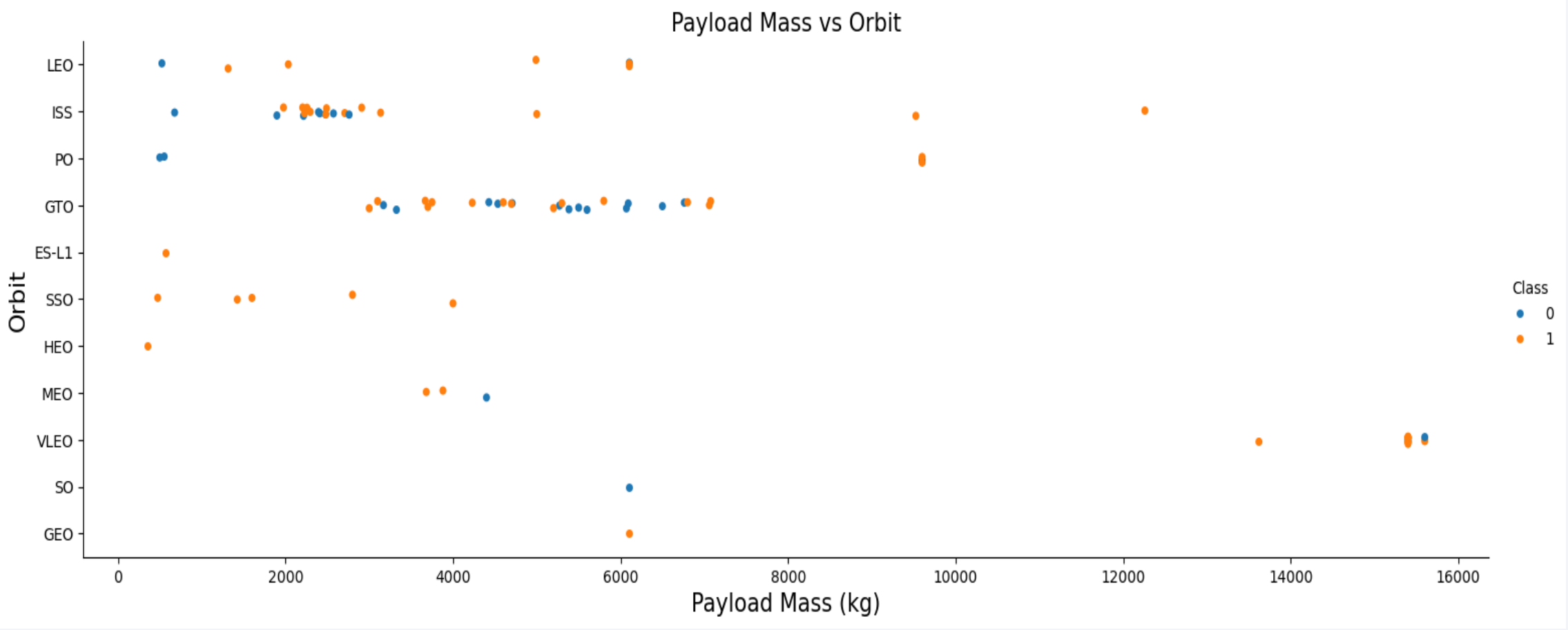
Flight Number vs. Orbit Type



- X-axis represents the Flight Number and Y-axis represents the Orbit Type. Each point represents a flight, positioned by orbit and flight number. Blue points indicate Class 0 (failures) while orange indicates successes.

- **Purpose:** The plot visualizes the distribution of flight numbers across different orbits and their corresponding success or failure outcomes.

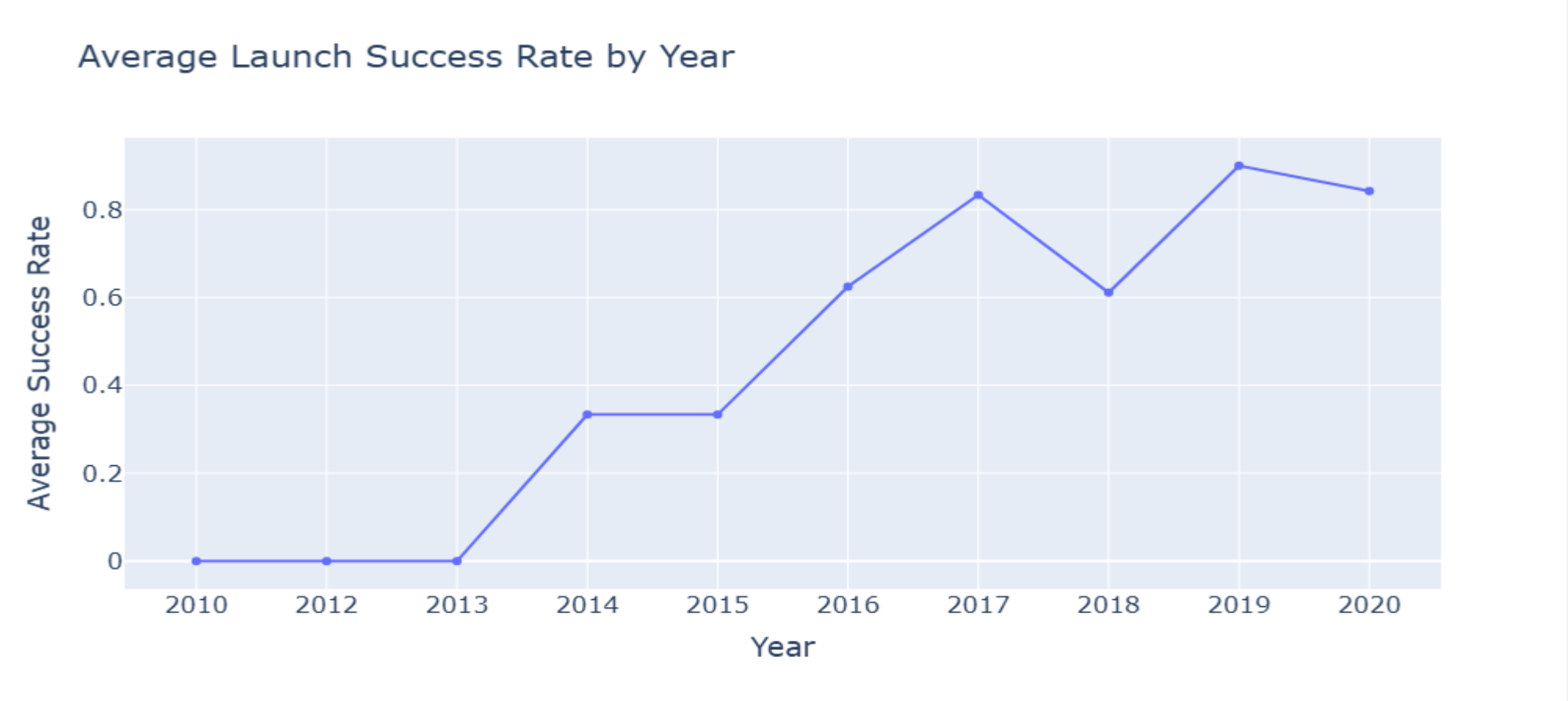
Payload vs. Orbit Type



- X-axis represents the Pay Load Mass and Y-axis represents the Orbit Type. Each point represents a flight, positioned by pay load mass and orbit type. Blue points indicate Class 0 (failures) while orange indicates successes.

- **Purpose:** The plot visualizes the distribution of pay load masses across different orbits and their corresponding success or failure outcomes. Examining this plot, we see that at higher pay load masses, **LEO, ISS, and PO** are more successful.

Launch Success Yearly Trend



- X-axis represents the years from 2010 to 2020 and Y-axis represents the Average Success Rate, ranging from 0 to 1.
- **Purpose:** Visualizes the success rate of missions for different years. Analyzing this data there is an upward trend in launch success rate over time, generally. Precisely, we can observe that the success rate since 2013 kept increasing till 2017 (stable in 2014) and after 2015 it started increasing.

All Launch Site Names

Launch Sites

- CCAFS SLC 40: Cape Canaveral Air Force Station Space Launch Complex 40.
- VAFB SLC 4E: Vandenberg Air Force Base Space Launch Complex 4E.
- KSC LC 39A: Kennedy Space Center Launch Complex 39A.
- CCAFS LC 40: Another complex at Cape Canaveral Air Force Station.

These launch sites are critical locations for various space missions, each contributing to the overall success of the space program.

Launch Site Names Begin with 'CCA'

Sql

```
# Execute the query to display 5 records where launch sites begin with 'CCA`
%sql SELECT  FROM SPACEXTABLE WHERE "Launch_Site" LIKE 'CCA%' LIMIT 5;
```

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

Sql

```
%%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
```

- The query first filters the records to include only those where the customer is "NASA (CRS)".
- Then, it sums up the payload masses of these filtered records to give the total payload mass.

Average Payload Mass by F9 v1.1

Sql

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

```
 Average_Payload_Mass
----------------------
2928.4
```

- **SELECT AVG("PAYLOAD_MASS__KG_") AS Average_Payload_Mass**: This part of the query calculates the average of the PAYLOAD_MASS__KG_ column and renames the result to Average_Payload_Mass.
- **FROM SPACEXTABLE**: Specifies the table from which the data is being queried.
- **WHERE "Booster_Version" = 'F9 v1.1'**: Filters the records to include only those where the Booster_Version column is 'F9 v1.1'.

First Successful Ground Landing Date

```
%%Sql
```

```
SELECT MIN("Launch_Date") AS First_Successful_Landing_Date
FROM SPACEXTABLE
WHERE "Landing_Outcome" = 'Success (ground pad)';
```

```
First_Successful_Landing_Date
-----------------------------
2015-12-22
```

- **SELECT MIN("Launch_Date") AS First_Successful_Landing_Date**: This part of the query finds the earliest date (minimum) in the Launch_Date column and renames the result to First_Successful_Landing_Date.
- **FROM SPACEXTABLE**: Specifies the table from which the data is being queried.
- **WHERE "Landing_Outcome" = 'Success (ground pad)'**: Filters the records to include only those with a successful landing outcome on a ground pad.

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_" > 4000
AND "PAYLOAD_MASS__KG_" < 6000;
```

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

- **SELECT "Booster_Version"**: This part of the query selects the names of the booster versions.
- **FROM SPACEXTABLE**: Specifies the table from which the data is being queried.
- **WHERE "Landing_Outcome" = 'Success (drone ship)'**: Filters the records to include only those where the landing outcome was successful on a drone ship.
- **AND "PAYLOAD_MASS__KG_" > 4000 AND "PAYLOAD_MASS__KG_" < 6000**: Further filters the records to include only those with a payload mass greater than 4000 kg but less than 6000 kg.

Total Number of Successful and Failure Mission Outcomes

%%Sql

```
SELECT  "Mission_Outcome", COUNT() AS Total_Count
FROM SPACEXTABLE
GROUP BY  "Mission_Outcome";
```

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

- ***SELECT "Mission_Outcome", COUNT()* AS Total_Count:** This part of the query selects the Mission_Outcome column and counts the number of occurrences of each outcome, renaming the count as Total_Count.
- **FROM SPACEXTABLE**: Specifies the table from which the data is being queried.
- **GROUP BY "Mission_Outcome"**: Groups the records by the Mission_Outcome column so that the counts are calculated for each unique outcome.

Boosters Carried Maximum Payload

Python and Sql

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

bList = [b_version[b][0] for b in range(len(b_version))]
print(bList)
```

```
['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 ']
```

- The query result shows that there are 12 booster versions that carried the maximum payload mass.
- This information is useful for identifying the most capable booster in terms of payload capacity, which can be crucial for planning future missions.
- This query helps pinpoint the boosters with the highest payload capacity, aiding in performance analysis and strategic planning

2015 Launch Records

%%sql

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

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

- This query helps in identifying specific instances of drone ship landing failures, providing insights into the performance and challenges faced during those missions in 2015.

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

%%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;
```

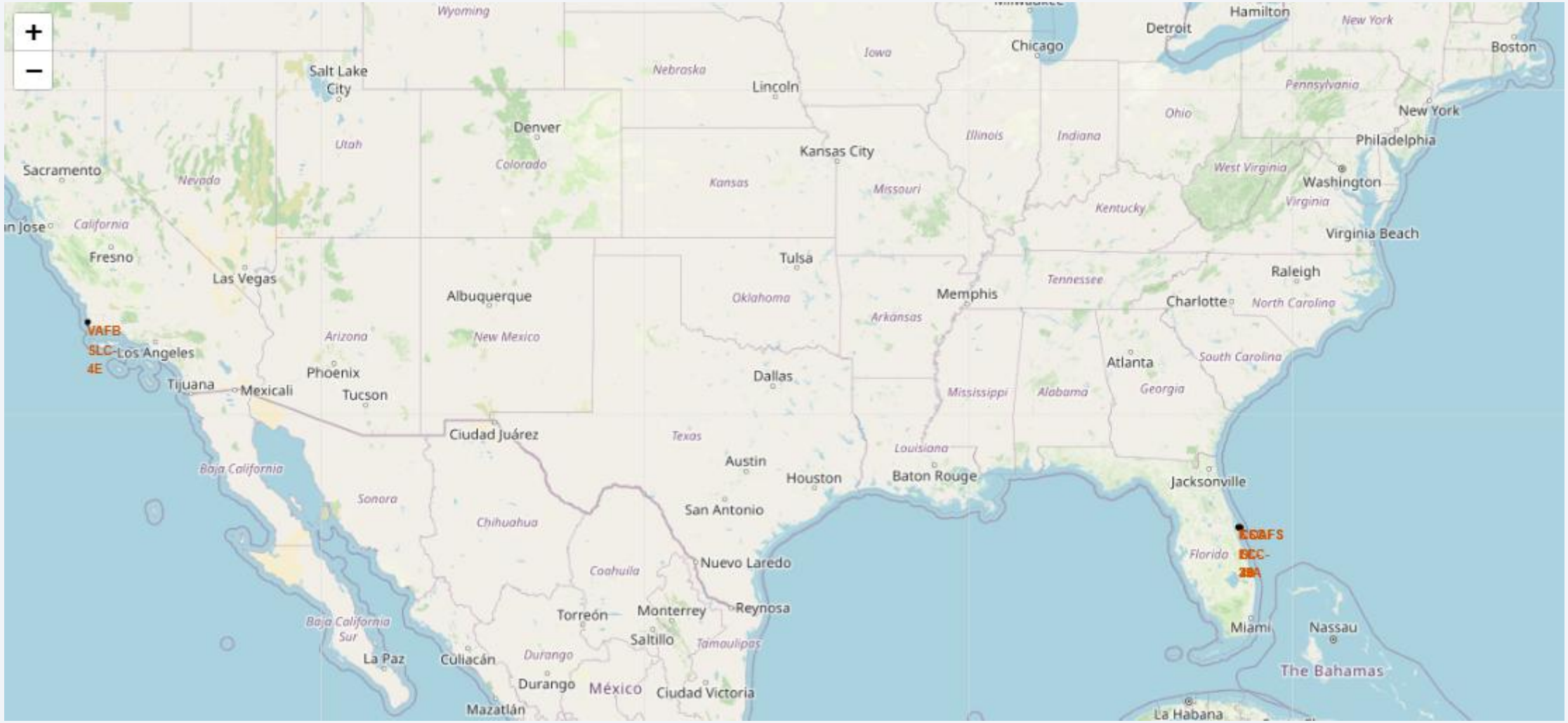
Landing_Outcome	Outcome_Count
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

This This ranking provides a clear understanding of the frequency of each landing outcome within the specified date range, helping to analyze the success rates and identify areas for potential improvement.

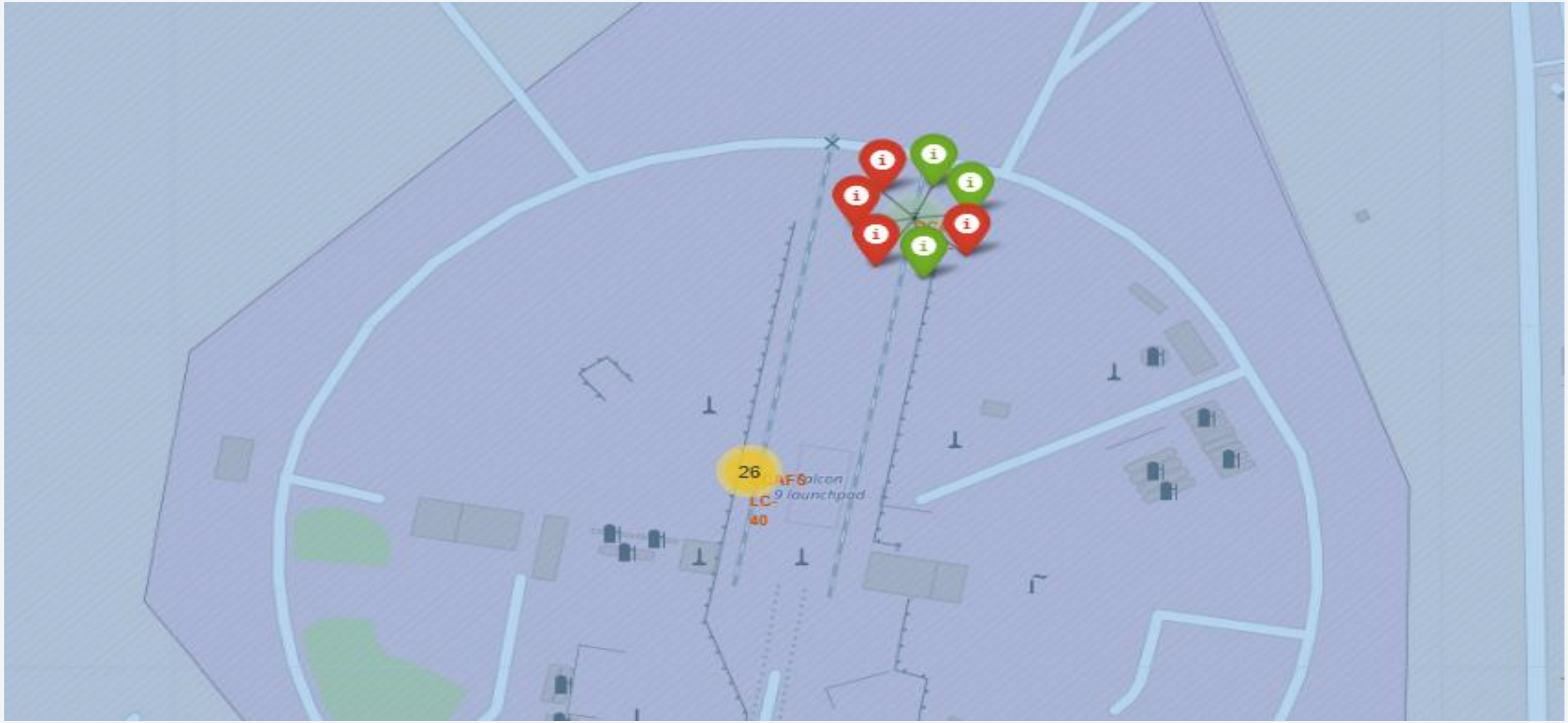
Section 3

Launch Sites Proximities Analysis

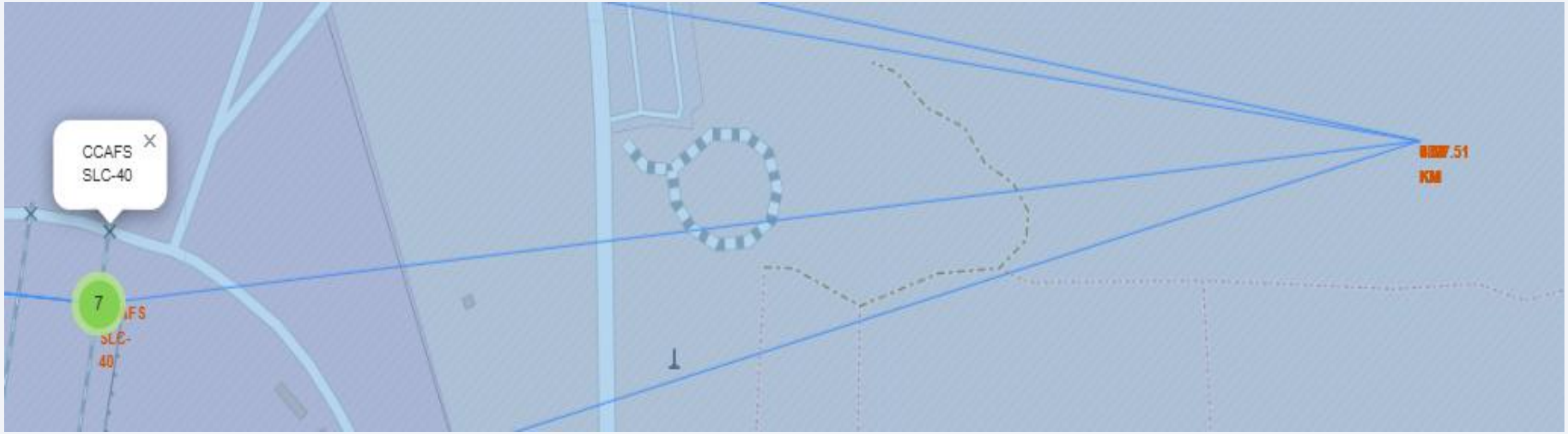
All Launch Sites on a Folium Map

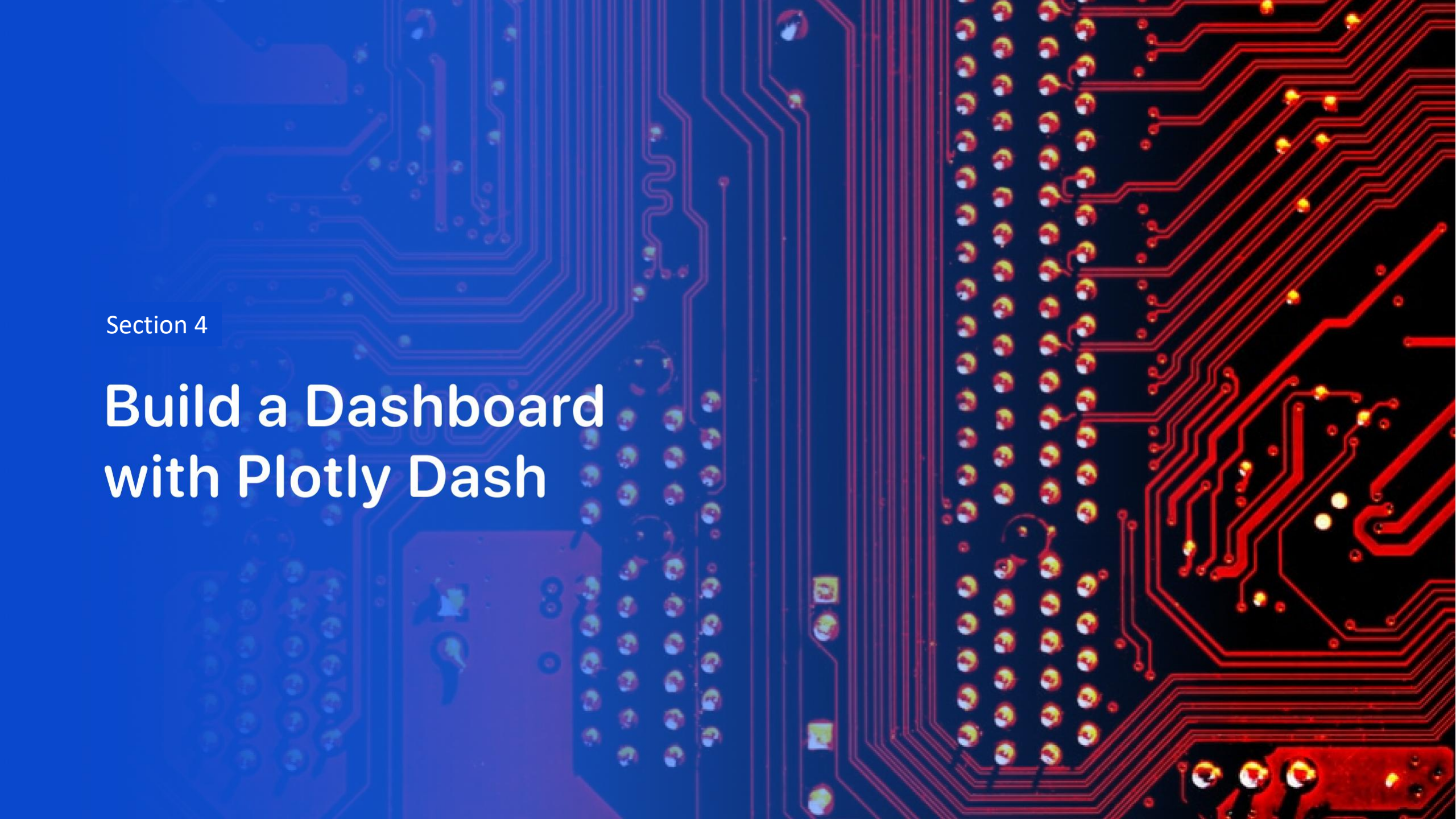


Launch Outcomes on a Folium Map

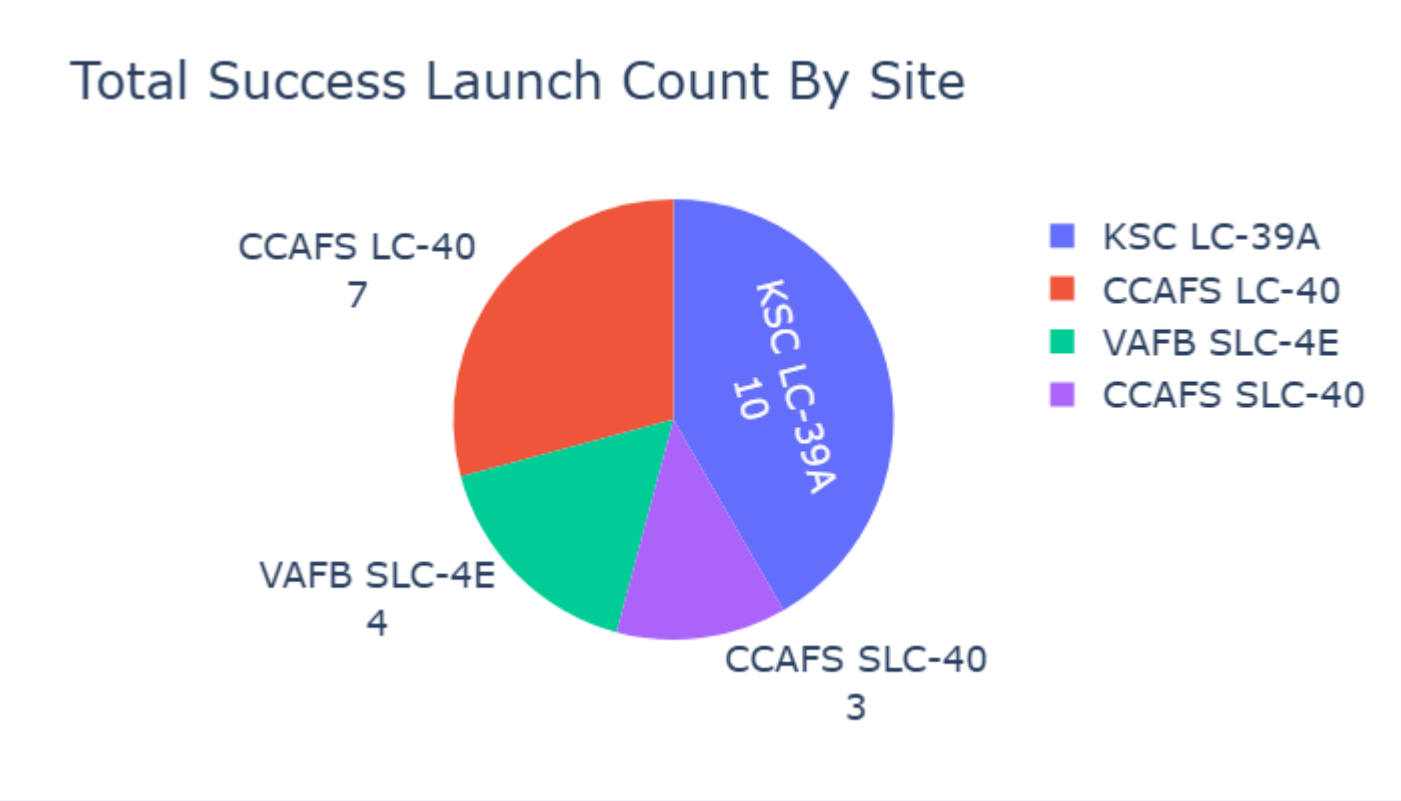


Selected Launch Site on a Folium Map




Section 4
Build a Dashboard with Plotly Dash

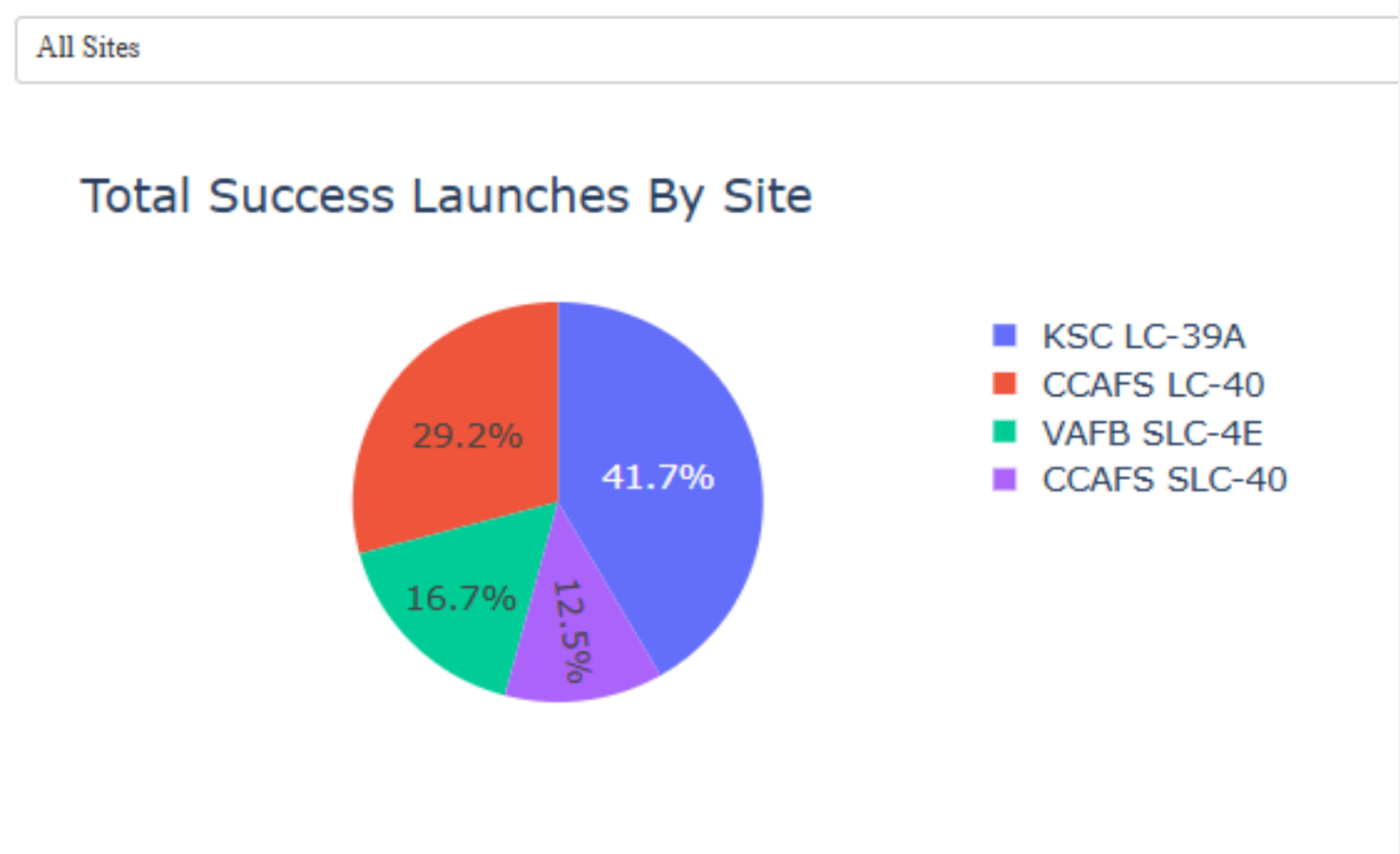
Launch Success Count for All Sites



Insight:

- The pie chart represents the distribution of successful launches across different sites.
- We see that the Launch Site with the highest success count is KSC LC-39A: 10

Launch Success Rate for All Sites



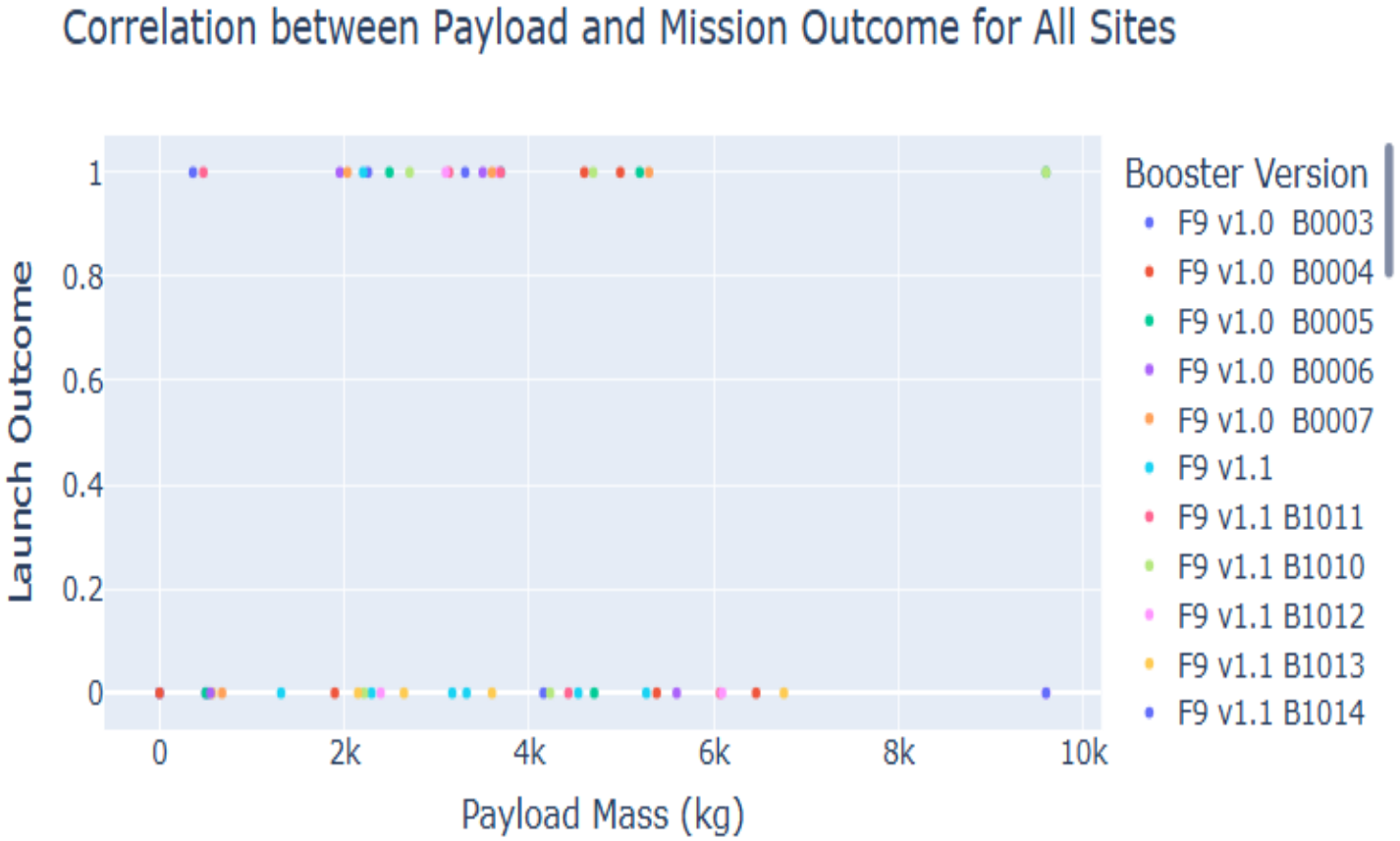
Insight:

- The pie chart represents the distribution of successful launches across different sites.
- We see that the Launch Site with the highest success rate is KSC LC-39A: 41.7%

Dashboard: Payload vs. Launch Outcome

Payload range (Kg):

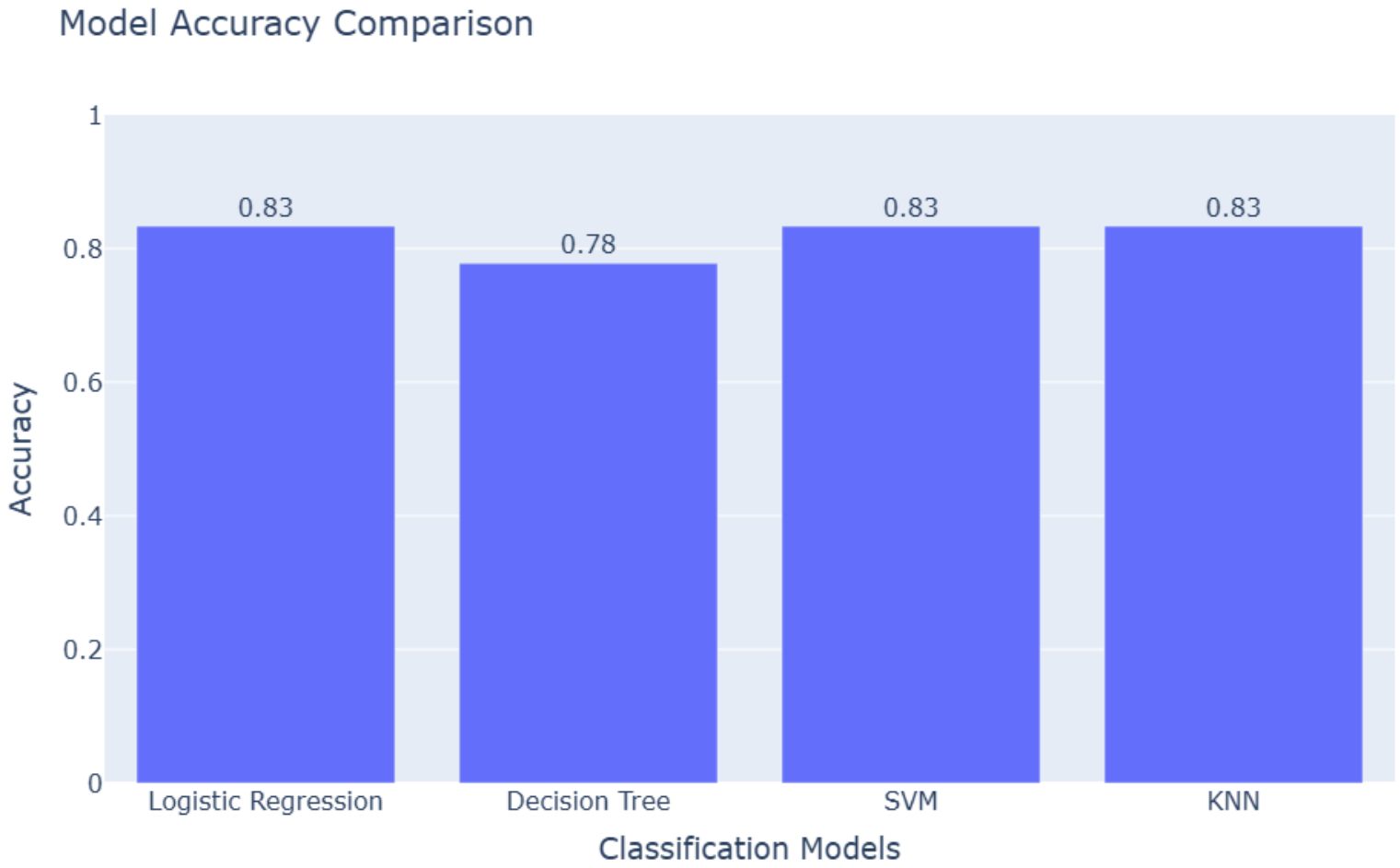
0 2500 5000 7500 10000



Section 5

Predictive Analysis (Classification)

Classification Accuracy

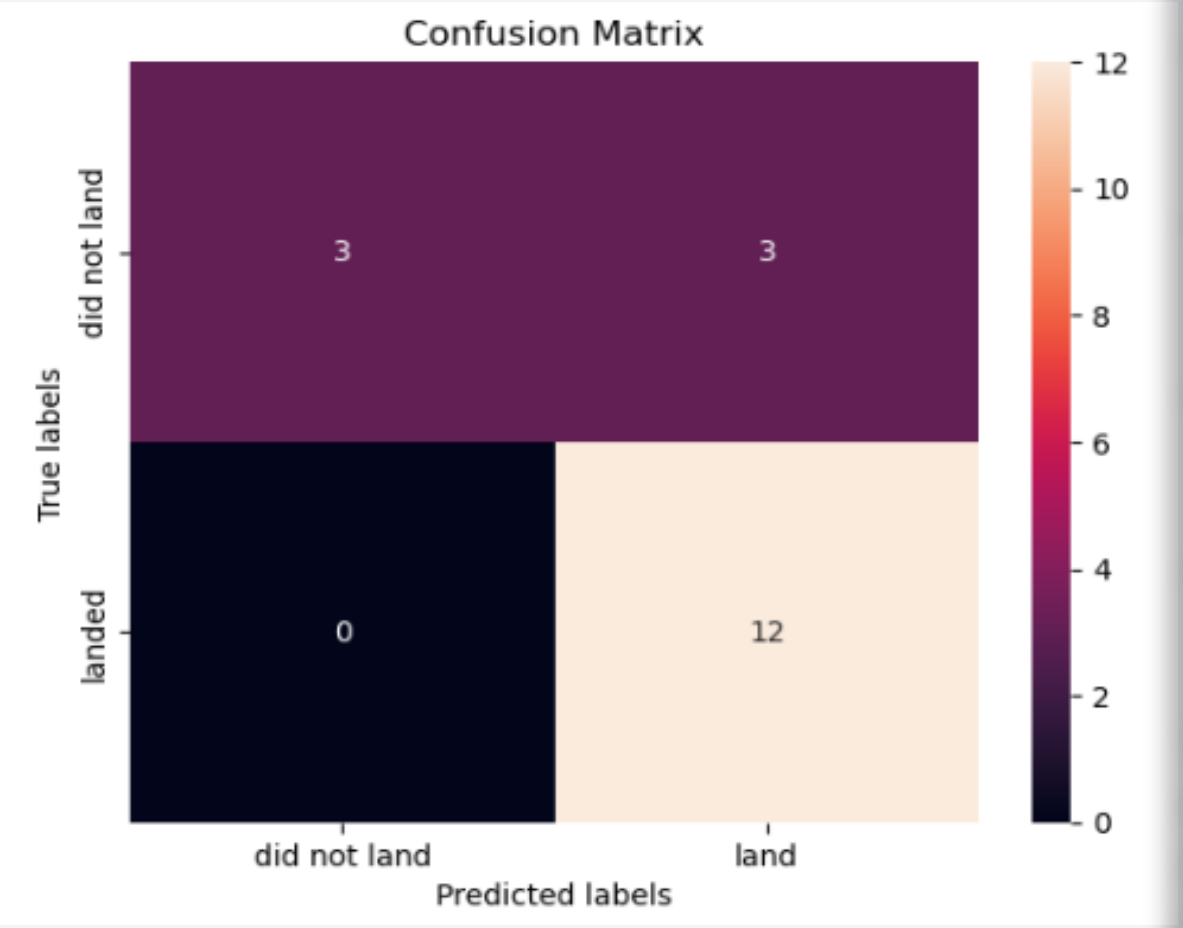


Model(s) with the Highest Accuracy

Logistic Regression, SVM, KNN

Accuracy: 0.83

Confusion Matrix



- True Positives (landed and predicted as landed): 12
- True Negatives (did not land and predicted as did not land): 3
- False Positives (did not land but predicted as landed): 3
- False Negatives (landed but predicted as did not land): 0

This visualization helps in understanding the distribution of correct and incorrect predictions and allows for a deeper analysis of the model's performance beyond just the accuracy score.

Conclusions

In our analysis, we adopted a comprehensive methodology to gain insights into SpaceX's launch data and predict launch outcomes:

Methodology

- Acquired data by making GET requests to the SpaceX API.
- Processed and cleaned the data for analysis.
- Conducted EDA using visualization and SQL to uncover key patterns and trends.
- Utilized Folium and Plotly Dash to create interactive maps and dashboards.
- Built and evaluated classification models to predict launch success.

Results

- Gained valuable insights into launch patterns and success rates.
- Analyzed the proximity of launch sites to key geographical features.
- Developed an interactive dashboard using Plotly Dash for real-time data visualization.
- Identified Logistic Regression, SVM, and KNN as top-performing models, each achieving an accuracy of 0.83.

Conclusions

- Through this multi-faceted approach, we enhanced our understanding of SpaceX's launch data, visualized key insights interactively, and accurately predicted launch outcomes using advanced classification models.
- These efforts underscore the potential of data science in driving strategic decisions and operational improvements.

Appendix

Falcon 9 first stage will land successfully

Appendix

```python
from graphviz import Digraph

dot = Digraph()

# Define the nodes with labels
dot.node('A', 'Data Preprocessing')
dot.node('B', 'Model Selection')
dot.node('C', 'Model Training')
dot.node('D', 'Model Evaluation')
dot.node('E', 'Model Improvement')
dot.node('F', 'Best Model Selection')

# Define the sub-nodes
dot.node('A1', 'Cleaned Data')
dot.node('A2', 'Feature Engineering')
dot.node('B1', 'Chosen Algorithms')
dot.node('C1', 'Train-Test Split')
dot.node('C2', 'Cross-Validation')
dot.node('D1', 'Metrics')
dot.node('D2', 'Confusion Matrix')
dot.node('E1', 'Hyperparameter Tuning')
dot.node('E2', 'Feature Scaling')
dot.node('E3', 'Ensemble Methods')
dot.node('F1', 'Comparative Analysis')
dot.node('F2', 'Final Model')
```

```python
# Connect the main phases from left to right
dot.edge('A', 'B')
dot.edge('B', 'C')
dot.edge('C', 'D')
dot.edge('D', 'E')
dot.edge('E', 'F')

# Connect the sub-nodes to their main phases
dot.edge('A', 'A1', style='dashed')
dot.edge('A', 'A2', style='dashed')
dot.edge('B', 'B1', style='dashed')
dot.edge('C', 'C1', style='dashed')
dot.edge('C', 'C2', style='dashed')
dot.edge('D', 'D1', style='dashed')
dot.edge('D', 'D2', style='dashed')
dot.edge('E', 'E1', style='dashed')
dot.edge('E', 'E2', style='dashed')
dot.edge('E', 'E3', style='dashed')
dot.edge('F', 'F1', style='dashed')
dot.edge('F', 'F2', style='dashed')

# Set graph direction from left to right
dot.attr(rankdir='LR')

# Save and render the graph
dot.render('model_development_flowchart', format='png')
```

Python code snippet used in generating the flowchart for model development in this project.

Appendix

```python
import pandas as pd
import plotly.express as px

# Filter data to include only successful launches
success_df = spacex_df[spacex_df['class'] == 1]

# Group by 'Launch Site' and count the number of successes
success_count = success_df.groupby('Launch Site').size().reset_index(name='count')

# Create a pie chart
fig = px.pie(success_count, values='count', names='Launch Site',
             title='Total Success Launch Count By Site',
             labels={'count': 'Number of Successful Launches'})

# Update trace to show counts instead of percentages
fig.update_traces(textinfo='label+value', text=success_count['count'])

# Update layout for better readability
fig.update_layout(
    font=dict(size=18),  # Increase font size
    height=400,          # Reduce figure height
    width=700            # Reduce figure width
)

# Show the pie chart
fig.show()
```

Python code snippet used in generating the Launch Success Rate for all sites.

Thank you!