IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Handsome Bongani
Nyoni
30 November 2024

Outline

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

Executive Summary

The analysis included data preprocessing, exploratory visualization, and the development of classification models, such as Logistic Regression, SVM (Support Vector Machine), Decision Tree, and K-Nearest Neighbors (KNN). Hyperparameter tuning was conducted using GridSearchCV, and model performance was assessed through accuracy metrics and confusion matrices. Among the models evaluated, SVM emerged as the best performer, achieving the highest accuracy. Insights gained from exploratory data analysis (EDA) and interactive Folium maps revealed trends in payloads, orbit types, and mission outcomes, offering valuable tools for optimizing SpaceX mission planning.

Introduction

The SpaceX Falcon 9 and Heavy rocket launches have revolutionized the aerospace industry by introducing reusable rocket technology, significantly reducing launch costs and increasing efficiency. SpaceX's mission to make space exploration more accessible has generated global interest and provided a wealth of data for analysis. This project leverages historical launch data to uncover insights into payload distribution, launch outcomes, and site-specific trends while building predictive models to determine the likelihood of successful landings.

- The primary goal of this analysis is to address key questions, such as:
- What are the characteristics of successful and unsuccessful launches?
- How do payload mass and orbit type influence mission outcomes?
- Which machine learning model can most accurately predict landing success?

Through a combination of web scraping, data processing, exploratory analysis, and machine learning, this project aims to provide actionable insights that support SpaceX's ongoing innovation and further advancements in the aerospace industry.

Section 1

Methodology

Methodology

Executive summary

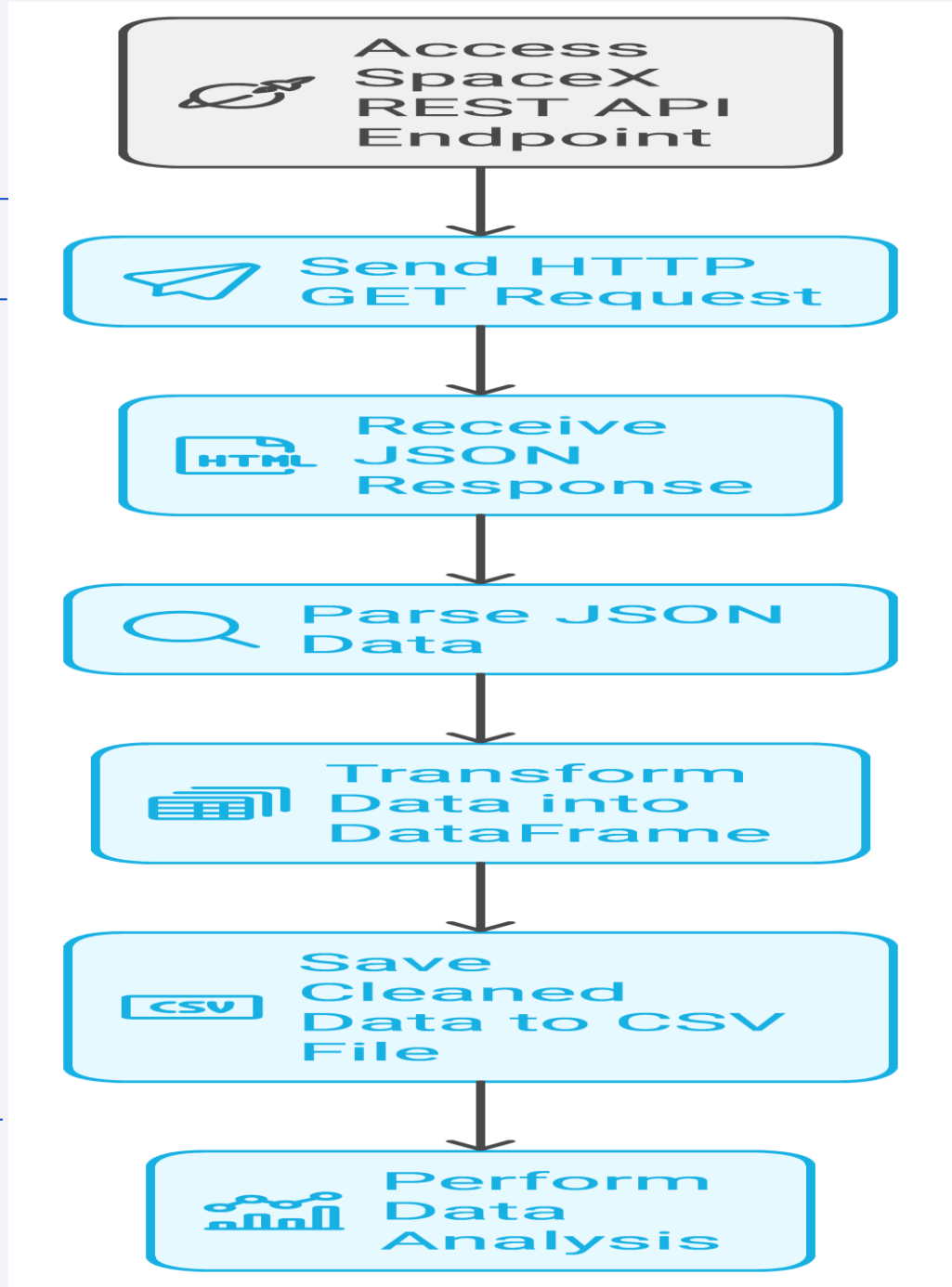
The project aimed to predict the success of Falcon 9 booster landings by systematically collecting, processing, and analyzing SpaceX launch data. Data was retrieved from the SpaceX REST API and historical repositories, focusing on key features such as launch sites, payload mass, orbits, and outcomes. Data wrangling involved cleaning missing values, removing duplicates, and engineering a binary Class column to represent landing success. Exploratory data analysis (EDA) was conducted using SQL and visualizations to uncover patterns and trends, including success rates by launch site and orbit type. Interactive analytics tools like Folium and Plotly Dash were employed to create dynamic geographical and statistical dashboards. Predictive analysis involved building classification models, including Logistic Regression, SVM, and Decision Trees, with hyperparameter tuning and performance evaluation using accuracy and F1-score metrics. The Decision Tree model achieved the highest accuracy, providing actionable insights for optimizing future SpaceX missions.

Data Collection

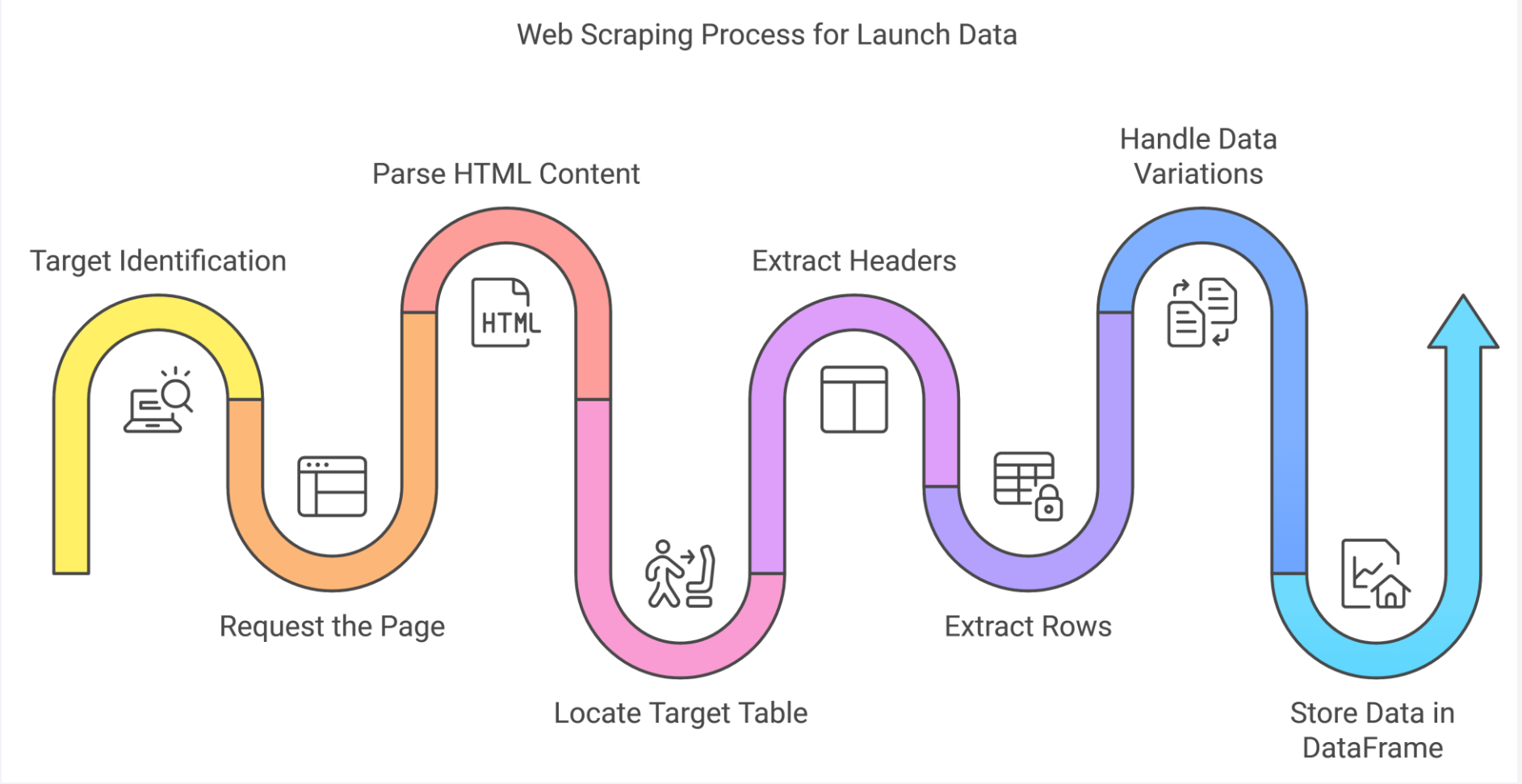
1. **Defining Objectives**:
   - Determine the scope: Collect data related to Falcon 9 launches, landing outcomes, and performance metrics.
   - Objective: Use the data to analyze launch success rates and predict future outcomes.
2. **Source Identification**:
   - SpaceX API for real-time launch data.
   - Historical datasets from SpaceX public repositories (e.g., CSV files provided in exercises).
3. **Data Retrieval**:
   - Use APIs for fetching real-time or programmatically accessible data.
   - Download historical datasets from trusted repositories.
4. **Data Cleaning and Preprocessing**:
   - Remove duplicates.
   - Handle missing values (e.g., imputation or exclusion).
   - Normalize formats for compatibility in analysis.

Data Collection – SpaceX API

- https://github.com/Hbnyoni/Handsome-Nyoni/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb

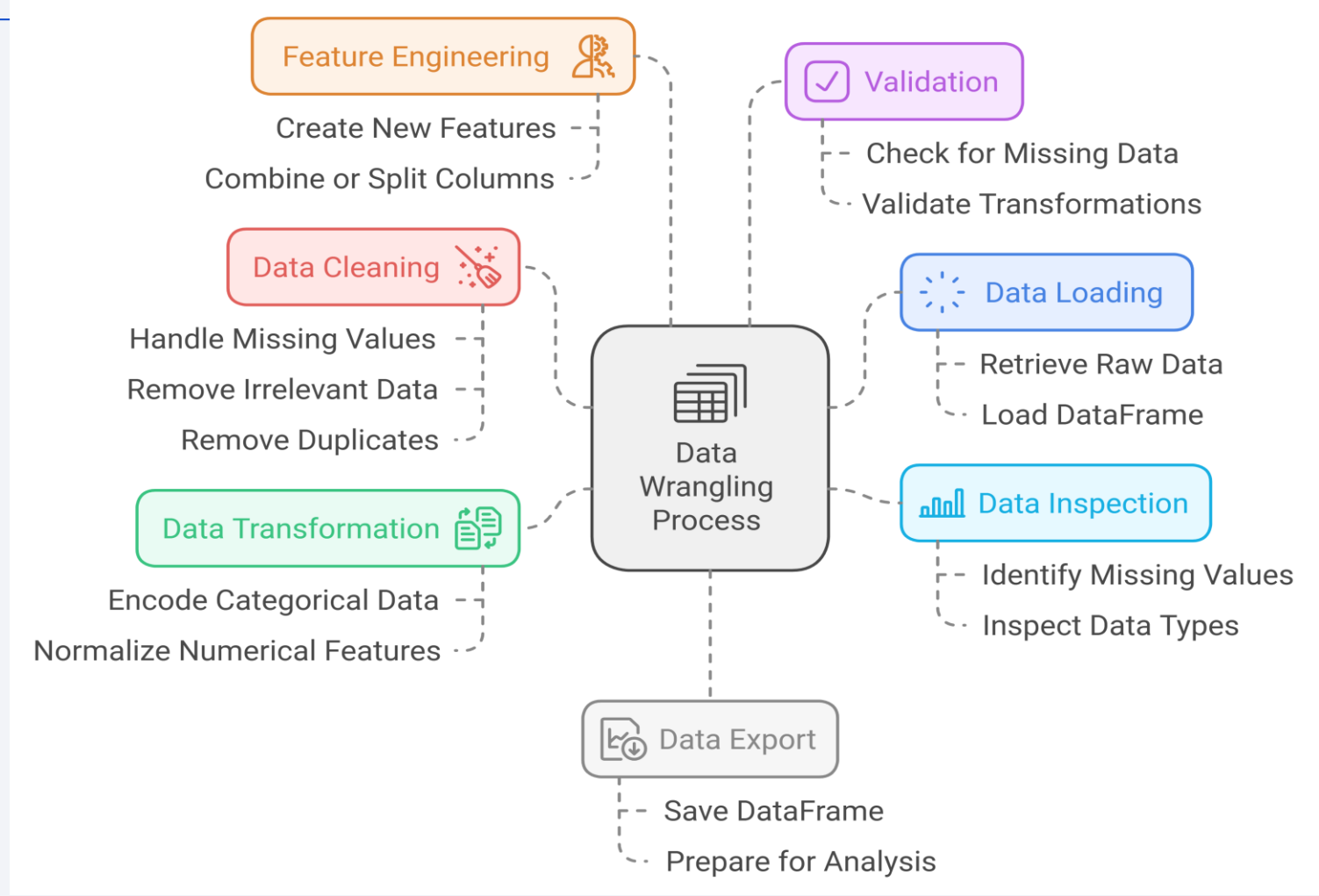


Data Collection - Scraping



https://github.com/Hbnyoni/Handsome-Nyoni/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb

Data Wrangling



https://github.com/Hbnyoni/Handsome-Nyoni/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

- **Scatter Plot: Payload Mass vs. Orbit**
- **Purpose**: To analyze the relationship between the payload mass and the orbit type.
- **Key Insight**:
  - Heavier payloads were more likely to land successfully in Polar, LEO, and ISS orbits.
  - For GTO orbit, there was no clear distinction between successful and unsuccessful landings, showing variability in outcomes.
- **Line Chart: Yearly Launch Success Trend**
- **Purpose**: To visualize the trend of launch success rates over the years.
- **Key Insight**:
  - This chart highlights how SpaceX's success rates improved over time, demonstrating learning and technological advancements in their landing strategies.

- https://github.com/Hbnyoni/Handsome-Nyoni/blob/main/edadataviz.ipynb

EDA with SQL

1.SELECT DISTINCT Launch_Site FROM SPACEXTABLE;
2.SELECT * FROM SPACEXTABLE WHERE Launch_Site LIKE 'CCA%' LIMIT 5;
3.SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTABLE WHERE Customer LIKE '%NASA (CRS)%';
4.SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTABLE WHERE Booster_Version = 'F9 v1.1';
5.SELECT MIN(Date) FROM SPACEXTABLE WHERE Landing_Outcome = 'Success (ground pad)';
6.SELECT Booster_Version FROM SPACEXTABLE WHERE Landing_Outcome = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000;
7.SELECT Mission_Outcome, COUNT(*) FROM SPACEXTABLE GROUP BY Mission_Outcome;
8.SELECT Booster_Version FROM SPACEXTABLE WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE);
9.SELECT SUBSTR(Date, 6, 2) AS Month, Booster_Version, Launch_Site FROM SPACEXTABLE WHERE SUBSTR(Date, 1, 4) = '2015' AND Landing_Outcome = 'Failure (drone ship)';

https://github.com/Hbnyoni/Handsome-Nyoni/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

Objects Added:

1.Markers:

1. Markers were placed at the coordinates of SpaceX launch sites.
2. Each marker included a popup to display information about the site (e.g., name, location).

2.Circles:

1. Circles were added to indicate a specific range or influence around a launch site.
2. These circles visually represented distances for proximity analysis.

3.Polylines (Lines):

1. Lines were drawn to connect launch sites to their nearest cities, railways, highways, or other significant landmarks.
2. This visualized the spatial relationship between the launch sites and key surrounding infrastructure.

4.Custom Icons:

1. Specific icons (e.g., railway symbols, highway signs) were used for better visualization and differentiation of map features.

5.MousePosition Plugin:

1. Enabled dynamic display of coordinates on the map to facilitate the manual addition of features like lines connecting launch sites to nearby landmarks.

https://github.com/Hbnyoni/Handsome-Nyoni/blob/main/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

•**Pie Chart:**
•A pie chart was added to display the success rate of SpaceX missions for all sites or a specific site.
•**Reason:** To provide a quick, visual representation of mission outcomes (success vs. failure) categorized by launch sites.
•**Scatter Plot:**
•A scatter plot was implemented to show the relationship between payload mass and mission outcomes.
•**Reason:** To explore whether heavier payloads correlate with the success or failure of landings across various orbit types.
•**Line Chart:**
•A line chart was added to visualize the yearly trend in mission success rates.
•**Reason:** To understand historical trends and the improvement of SpaceX's launch success over time.

Predictive Analysis (Classification)

- **Plots/Graphs Added:**

1. Pie Chart:

1. A pie chart was added to display the success rate of SpaceX missions for all sites or a specific site.
2. **Reason:** To provide a quick, visual representation of mission outcomes (success vs. failure) categorized by launch sites.

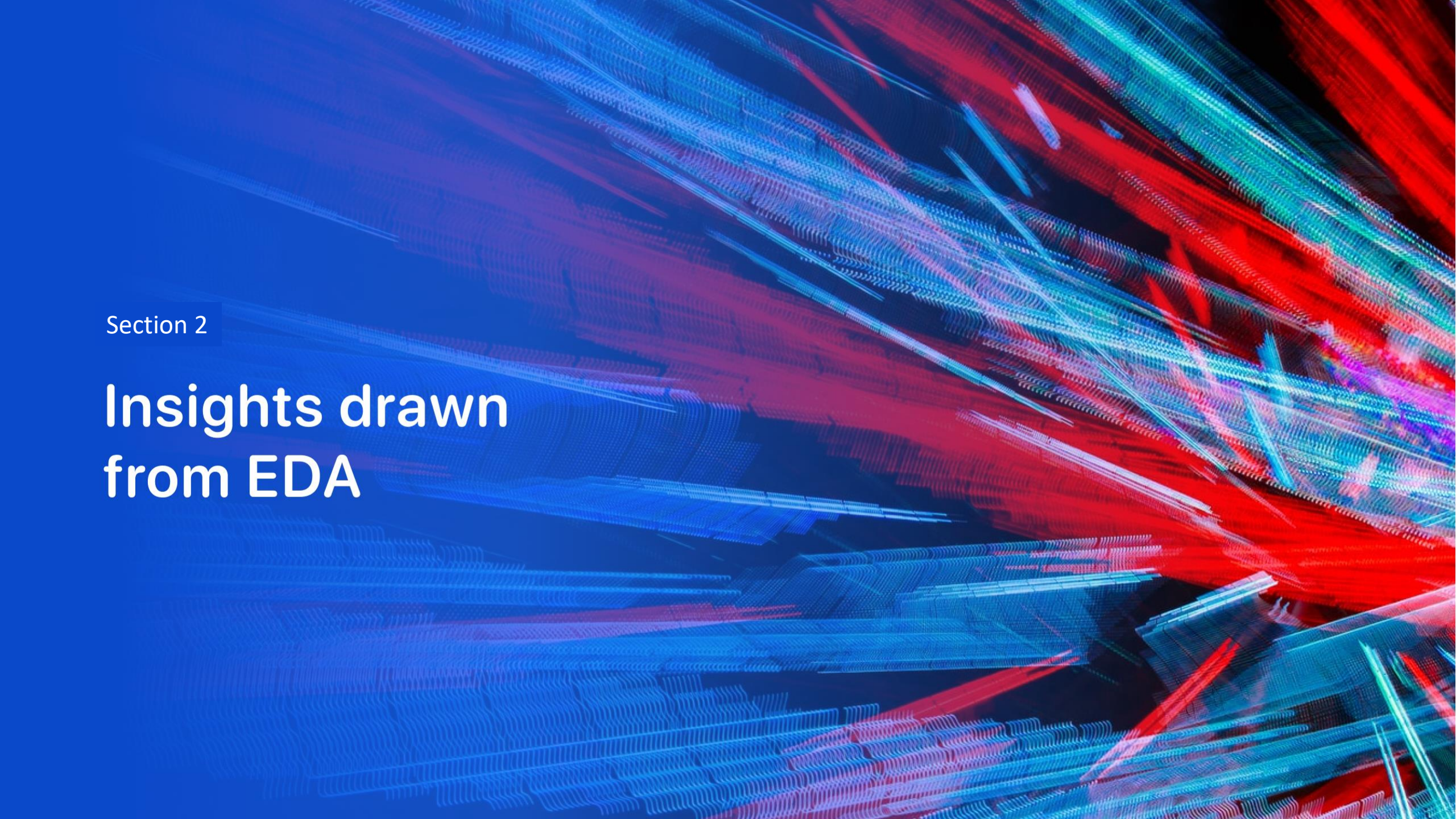
2. Scatter Plot:

1. A scatter plot was implemented to show the relationship between payload mass and mission outcomes.
2. **Reason:** To explore whether heavier payloads correlate with the success or failure of landings across various orbit types.

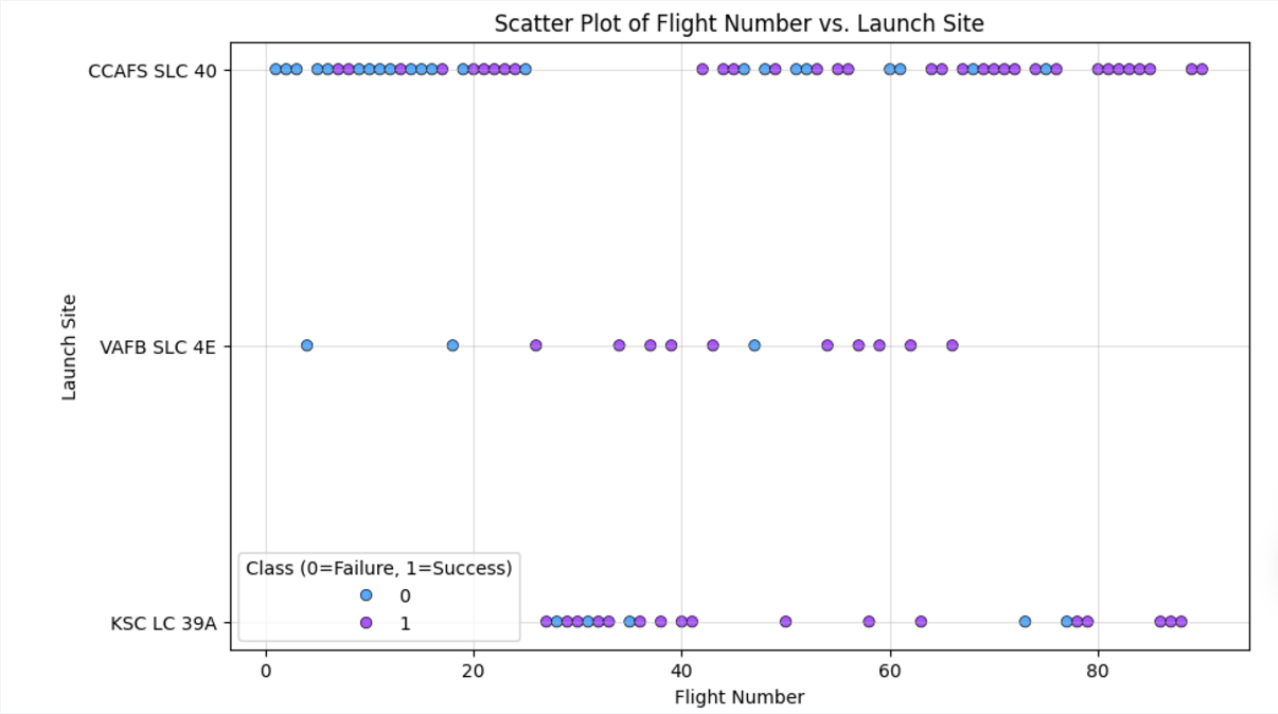
3. Line Chart:

1. A line chart was added to visualize the yearly trend in mission success rates.
2. **Reason:** To understand historical trends and the improvement of SpaceX's launch success over time.

https://github.com/Hbnyoni/Handsome-Nyoni/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

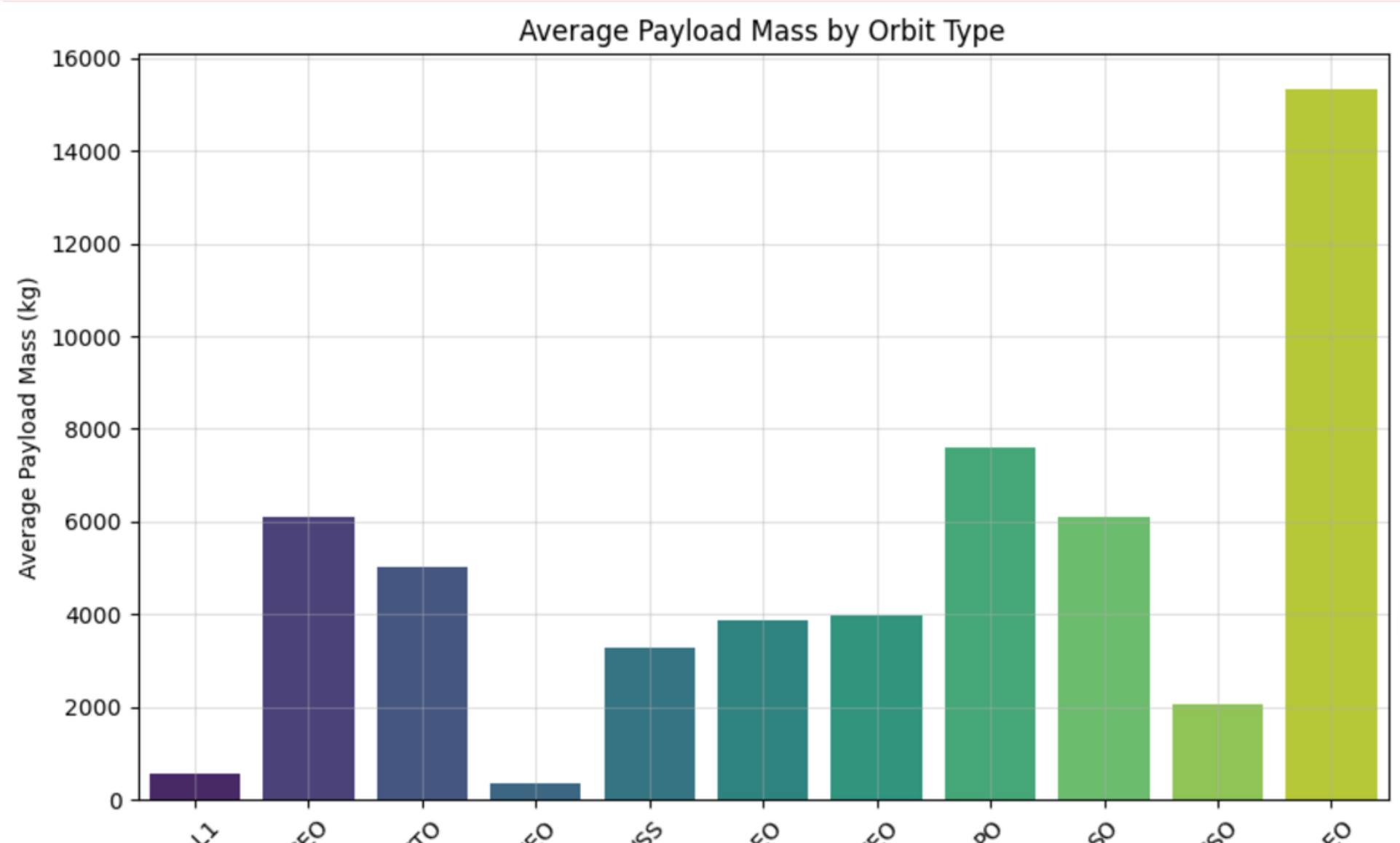

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site

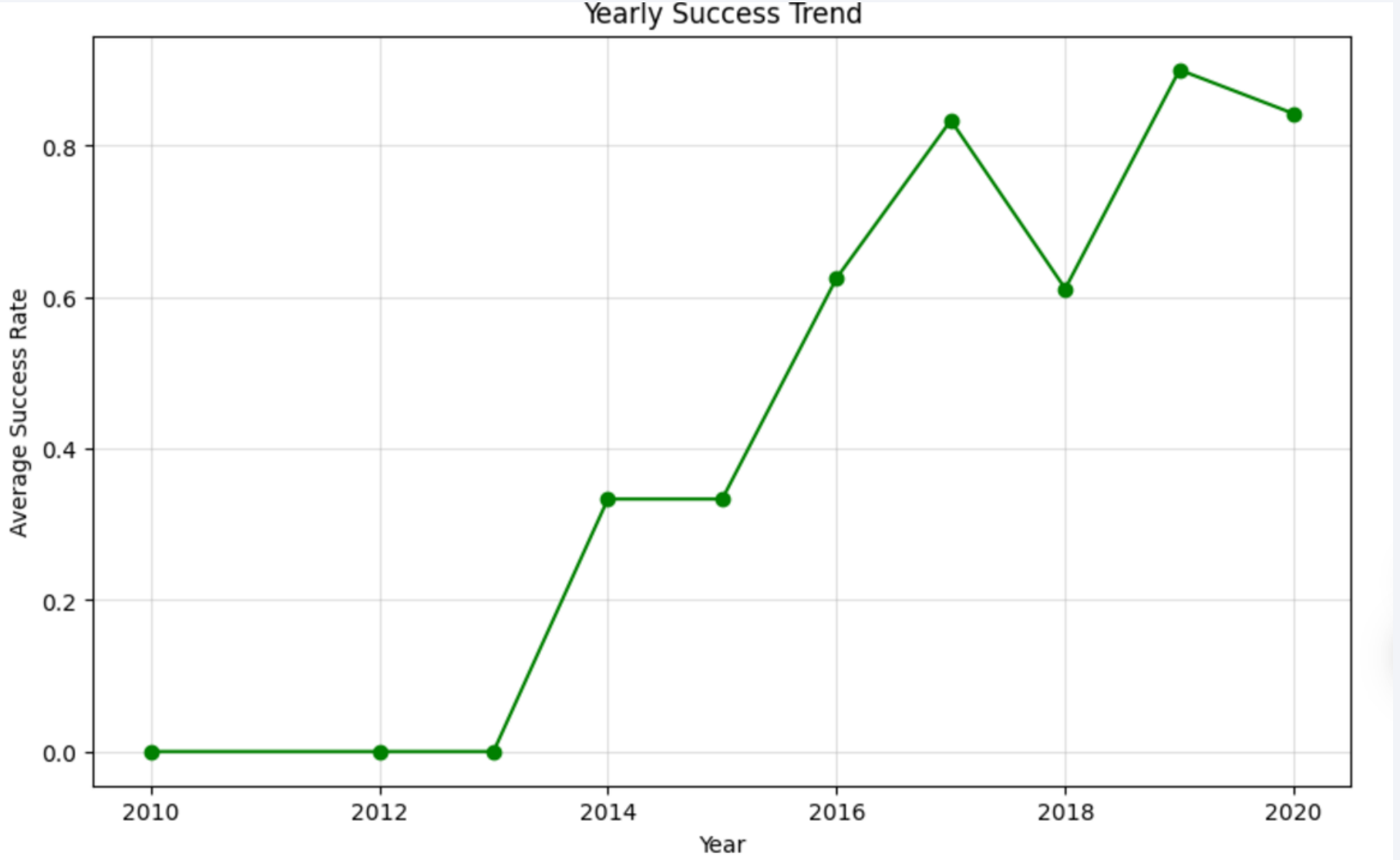



```
Average Payload Mass by Orbit Type:
Orbit
ES-L1      570.000000
GEO       6104.959412
GTO       5011.994444
HEO        350.000000
ISS       3279.938095
LEO       3882.839748
MEO       3987.000000
PO        7583.666667
SO        6104.959412
SSO       2060.000000
VLEO     15315.714286
Name: PayloadMass, dtype: float64
```

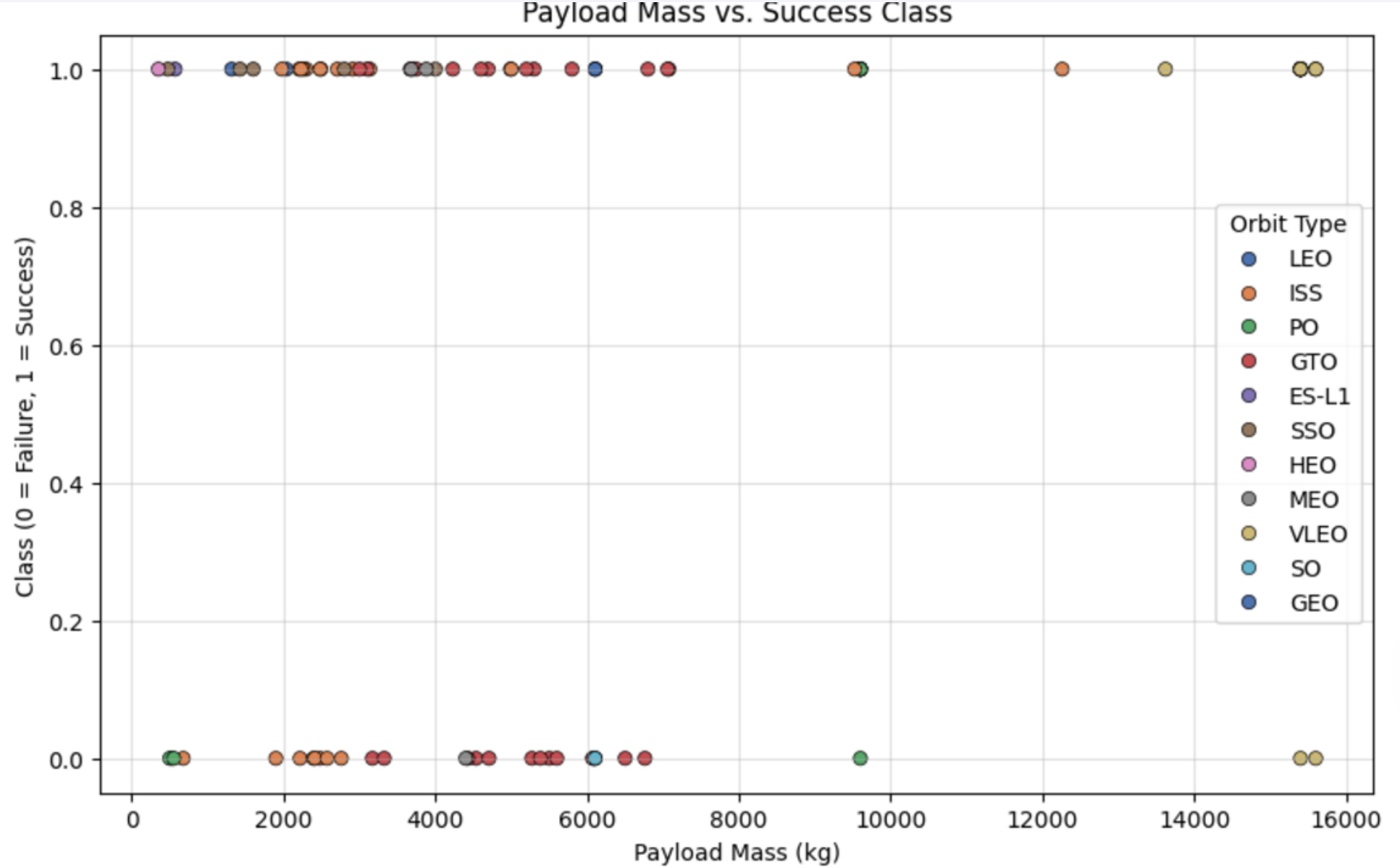
Payload vs. Launch Site



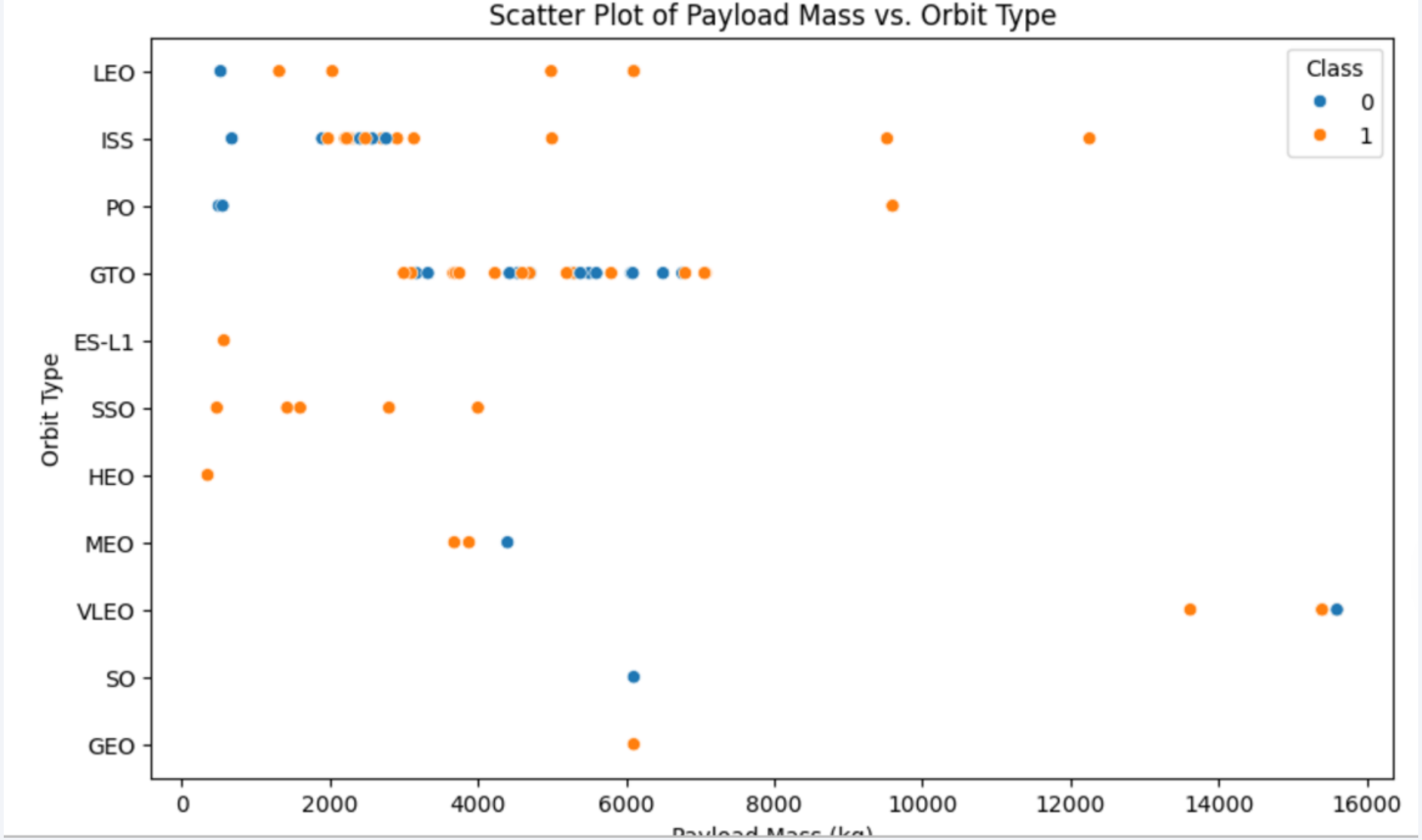
Success Rate vs. Orbit Type



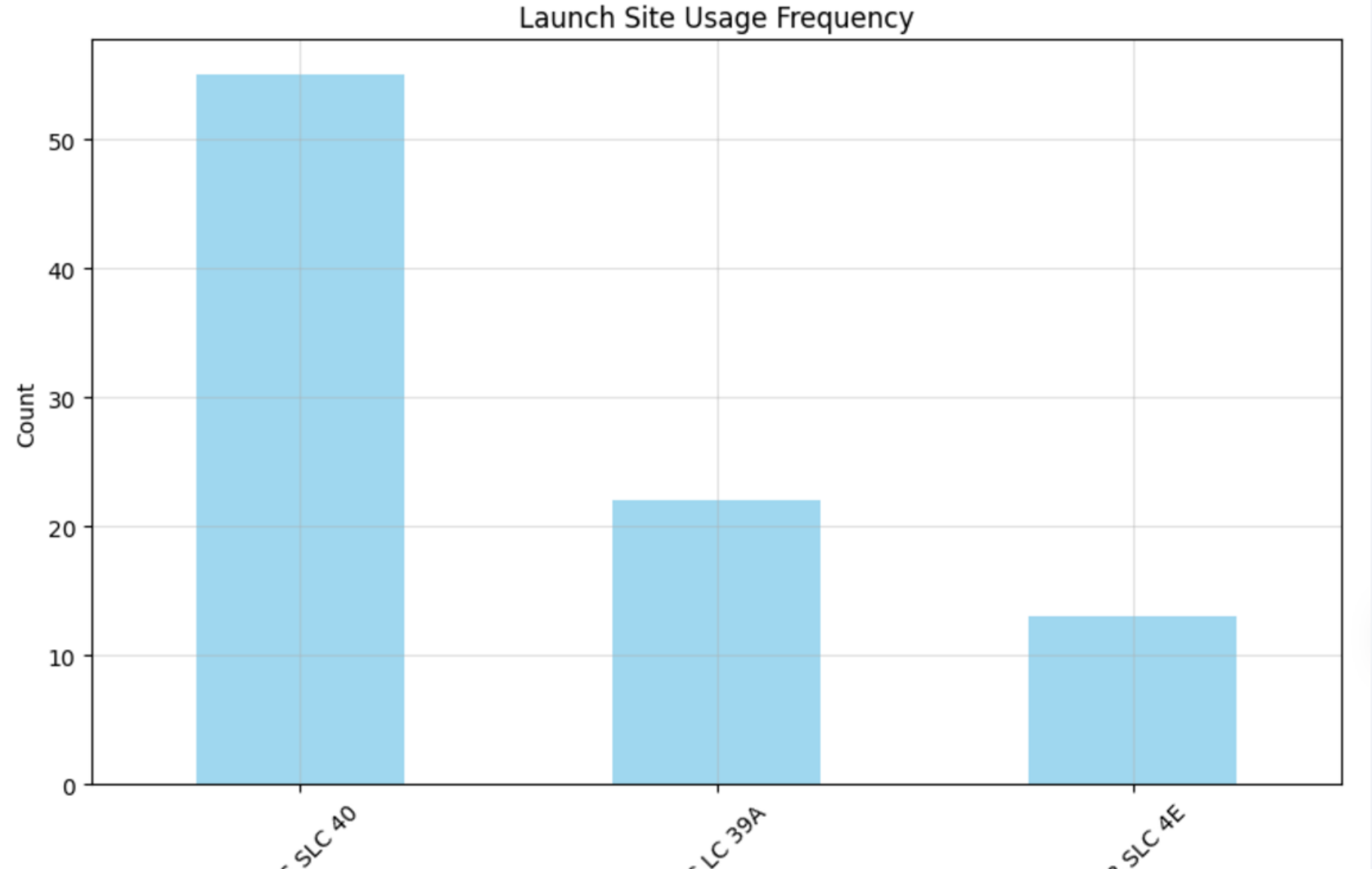
Flight Number vs. Orbit Type



Payload vs. Orbit Type



Launch Success Yearly Trend



All Launch Site Names

[6]:
```
# Find unique launch sites
unique_launch_sites = df['LaunchSite'].unique()

# Print the unique launch sites
print("Unique Launch Sites:")
for site in unique_launch_sites:
    print(site)
```

```
Unique Launch Sites:
CCAFS SLC 40
VAFB SLC 4E
KSC LC 39A
```

Launch Site Names Begin with 'CCA'

[8]:

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False
4	5	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1	False	False	False
5	6	2014-01-06	Falcon 9	3325.000000	GTO	CCAFS SLC 40	None None	1	False	False	False

Average Payload Mass by F9 v1.1

[10]:
```
# Filter records for the booster version F9 v1.1
f9_v1_1_payloads = df[df['BoosterVersion'] == 'F9 v1.1']


# Calculate the average payload mass
average_payload_mass = f9_v1_1_payloads['PayloadMass'].mean()


print(f"The average payload mass carried by booster version F9 v1.1 is: {average_payload_mass:.2f} kg")
```

```
The average payload mass carried by booster version F9 v1.1 is: nan kg
```

First Successful Ground Landing Date

[11]:
```
# Filter rows where Outcome indicates a successful landing on a ground pad
ground_pad_success = df[df['Outcome'] == 'True RTLS']

# Find the earliest date of the successful ground pad landing
first_success_date = ground_pad_success['Date'].min()

print(f"The date of the first successful landing outcome on a ground pad is: {first_success_date}")
```

```
The date of the first successful landing outcome on a ground pad is: 2015-12-22
```

Successful Drone Ship Landing with Payload between 4000 and 6000

[12]:
```
# Filter rows where the landing outcome is a successful drone ship landing and payload mass is within the
successful_boosters = df[
    (df['Outcome'] == 'True ASDS') &
    (df['PayloadMass'] > 4000) &
    (df['PayloadMass'] < 6000)
]

# Extract the unique booster names
booster_names = successful_boosters['BoosterVersion'].unique()

print("Boosters that successfully landed on a drone ship with payload mass between 4000 and 6000:")
for booster in booster_names:
    print(booster)
```

```
Boosters that successfully landed on a drone ship with payload mass between 4000 and 6000:
Falcon 9
```

Total Number of Successful and Failure Mission Outcomes

[13]:
```
# Grouping the outcomes and counting the occurrences
outcome_counts = df['Outcome'].value_counts()

# Separating success and failure outcomes
success_count = outcome_counts.get('True ASDS', 0) + outcome_counts.get('True RTLS', 0)
failure_count = len(df) - success_count  # Total rows minus success counts

# Displaying results
print("Total number of successful mission outcomes:", success_count)
print("Total number of failure mission outcomes:", failure_count)
```

```
Total number of successful mission outcomes: 55
Total number of failure mission outcomes: 35
```

Boosters Carried Maximum Payload

[14]:
```
# Find the maximum payload mass
max_payload_mass = df['PayloadMass'].max()

# Filter the rows with the maximum payload mass
boosters_with_max_payload = df[df['PayloadMass'] == max_payload_mass]

# Extract the booster names
booster_names = boosters_with_max_payload['BoosterVersion'].unique()

# Display the results
print("Boosters that carried the maximum payload mass:")
for booster in booster_names:
    print(booster)
```

```
Boosters that carried the maximum payload mass:
Falcon 9
```

2015 Launch Records

[15]:
```
# Filter the dataset for the year 2015
df['Year'] = pd.to_datetime(df['Date']).dt.year
data_2015 = df[df['Year'] == 2015]

# Further filter for failed landing outcomes on drone ships
failed_drone_ship = data_2015[(data_2015['Outcome'].str.contains('False ASDS', na=False))]

# Select relevant columns: Landing Outcome, Booster Version, and Launch Site
result = failed_drone_ship[['Outcome', 'BoosterVersion', 'LaunchSite']]

# Display the result
print("Failed landing outcomes on drone ships in 2015:")
print(result)
```

```
Failed landing outcomes on drone ships in 2015:
       Outcome BoosterVersion    LaunchSite
11  False ASDS       Falcon 9  CCAFS SLC 40
13  False ASDS       Falcon 9  CCAFS SLC 40
```

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

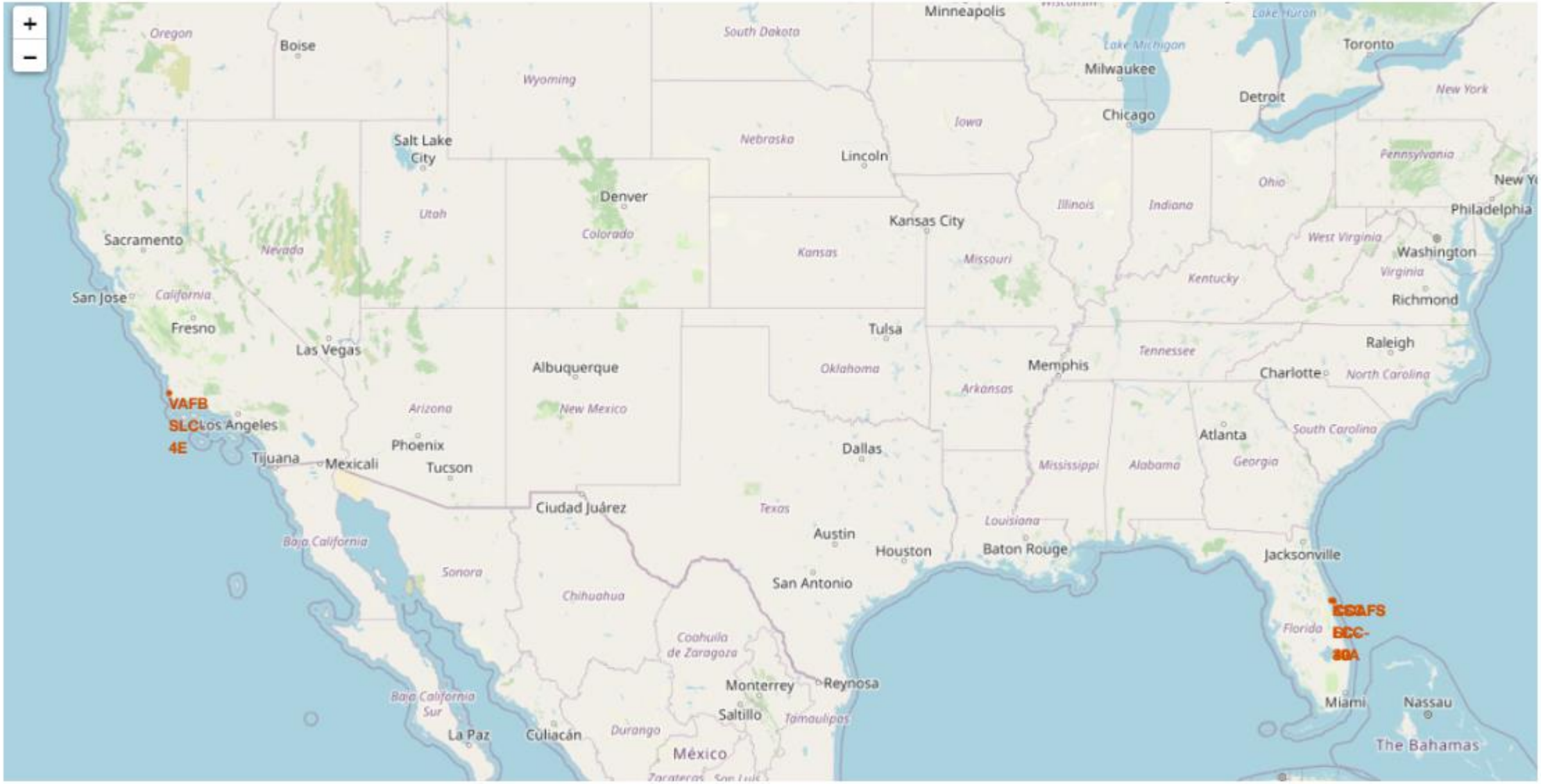
[16]:
```
# Filter the dataset for the specified date range
date_filtered_df = df[(pd.to_datetime(df['Date']) >= '2010-06-04') & (pd.to_datetime(df['Date']) <= '20

# Group by the Outcome column and count the occurrences
landing_outcomes_ranked = date_filtered_df['Outcome'].value_counts().reset_index()
landing_outcomes_ranked.columns = ['Outcome', 'Count']

# Sort the outcomes in descending order of their count
landing_outcomes_ranked_sorted = landing_outcomes_ranked.sort_values(by='Count', ascending=False)

# Display the result
print("Landing outcomes ranked by count (2010-06-04 to 2017-03-20):")
print(landing_outcomes_ranked_sorted)
```

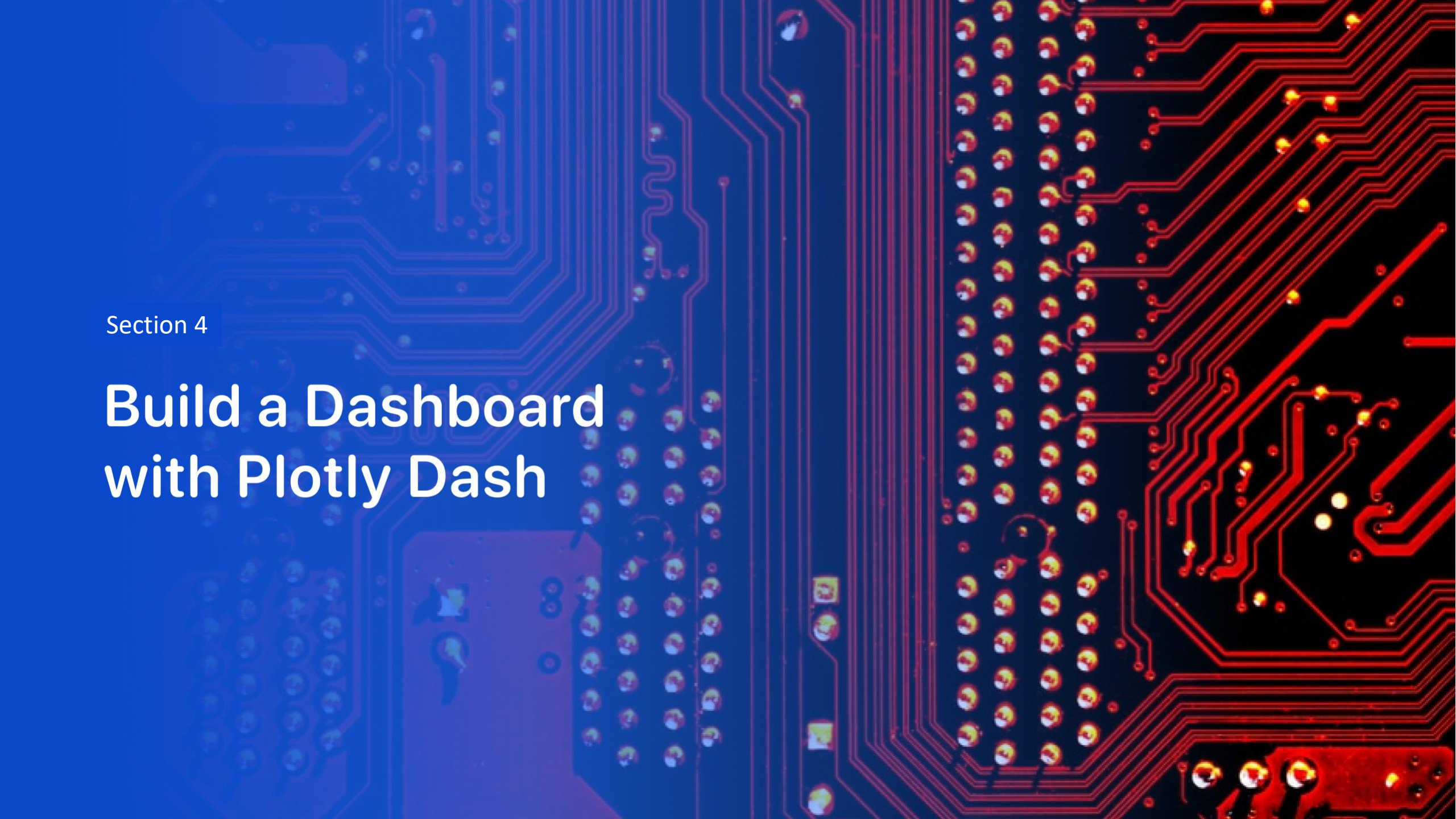
```
Landing outcomes ranked by count (2010-06-04 to 2017-03-20):
       Outcome  Count
0    None None      9
1     True ASDS      5
2    False ASDS      4
3     True Ocean      3
4     True RTLS      3
5   False Ocean      2
6     None ASDS      2
```

Section 3

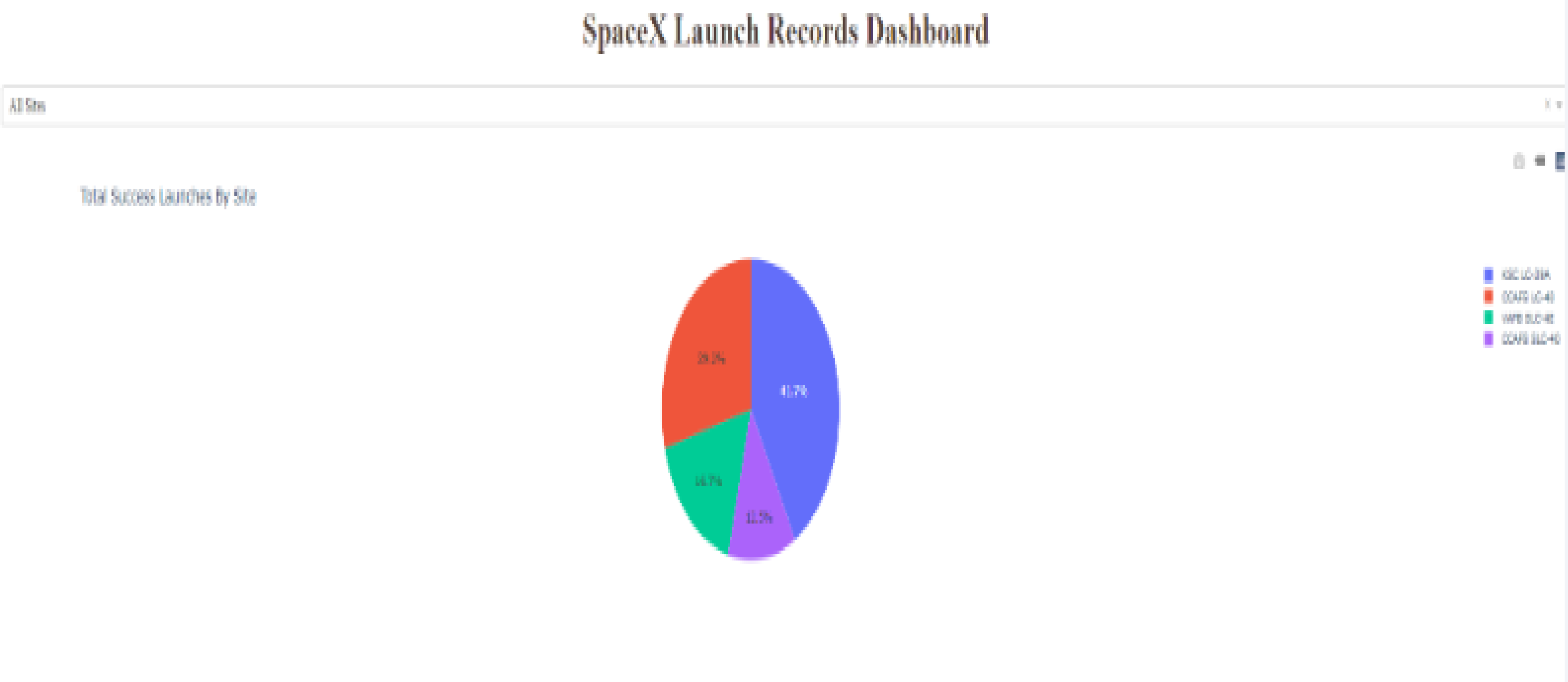
Launch Sites Proximities Analysis

<Folium Map Screenshot 1>

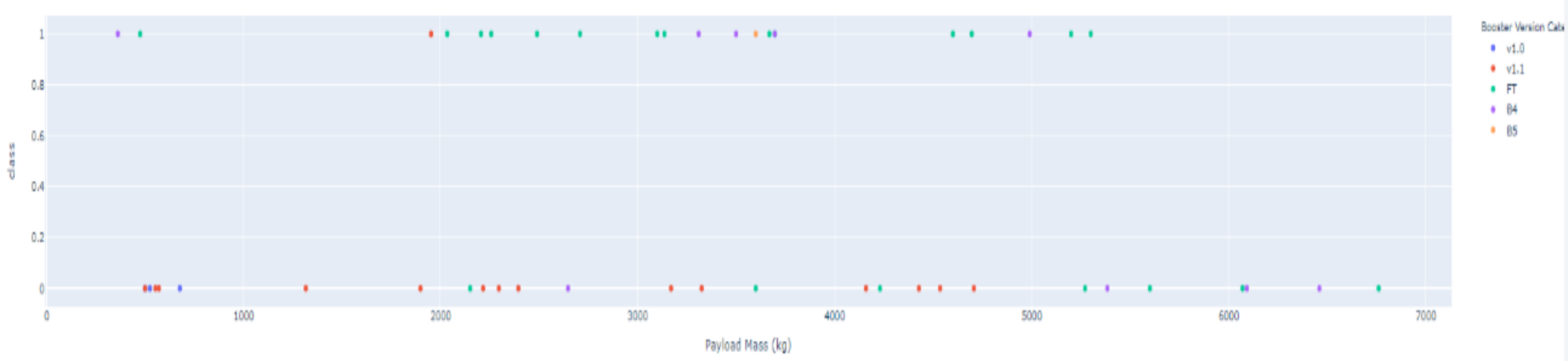



Section 4
Build a Dashboard with Plotly Dash

Dashboard


SpaceX Launch Records Dashboard
All Sites
Total Success Launches By Site

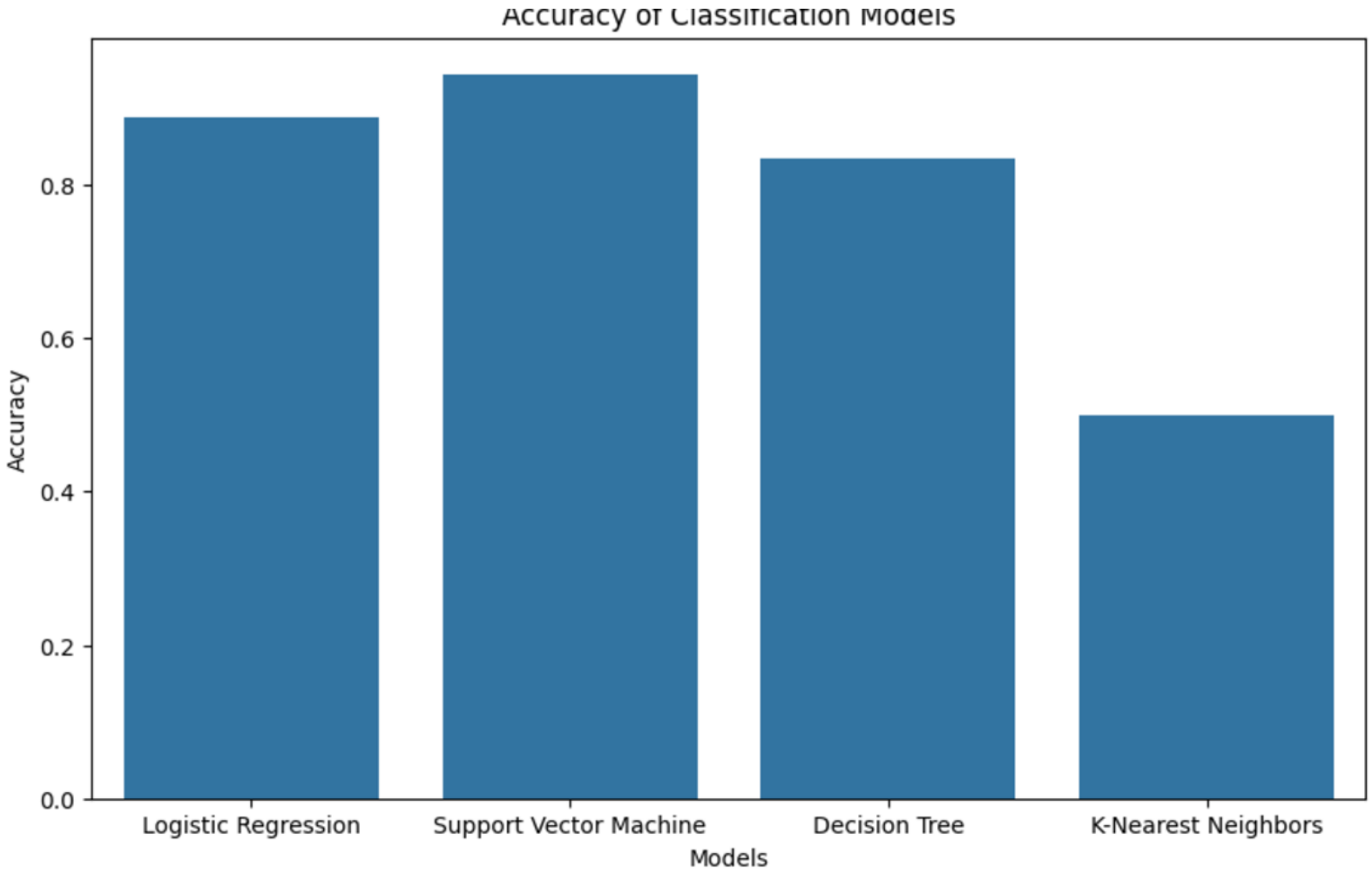
Dashboard


class
1
0.8
0.6
0.4
0.2
0
0
1000
2000
3000
4000
5000
6000
7000
Payload Mass (kg)
v1.0
v1.1
FT
B4
B5

Section 5

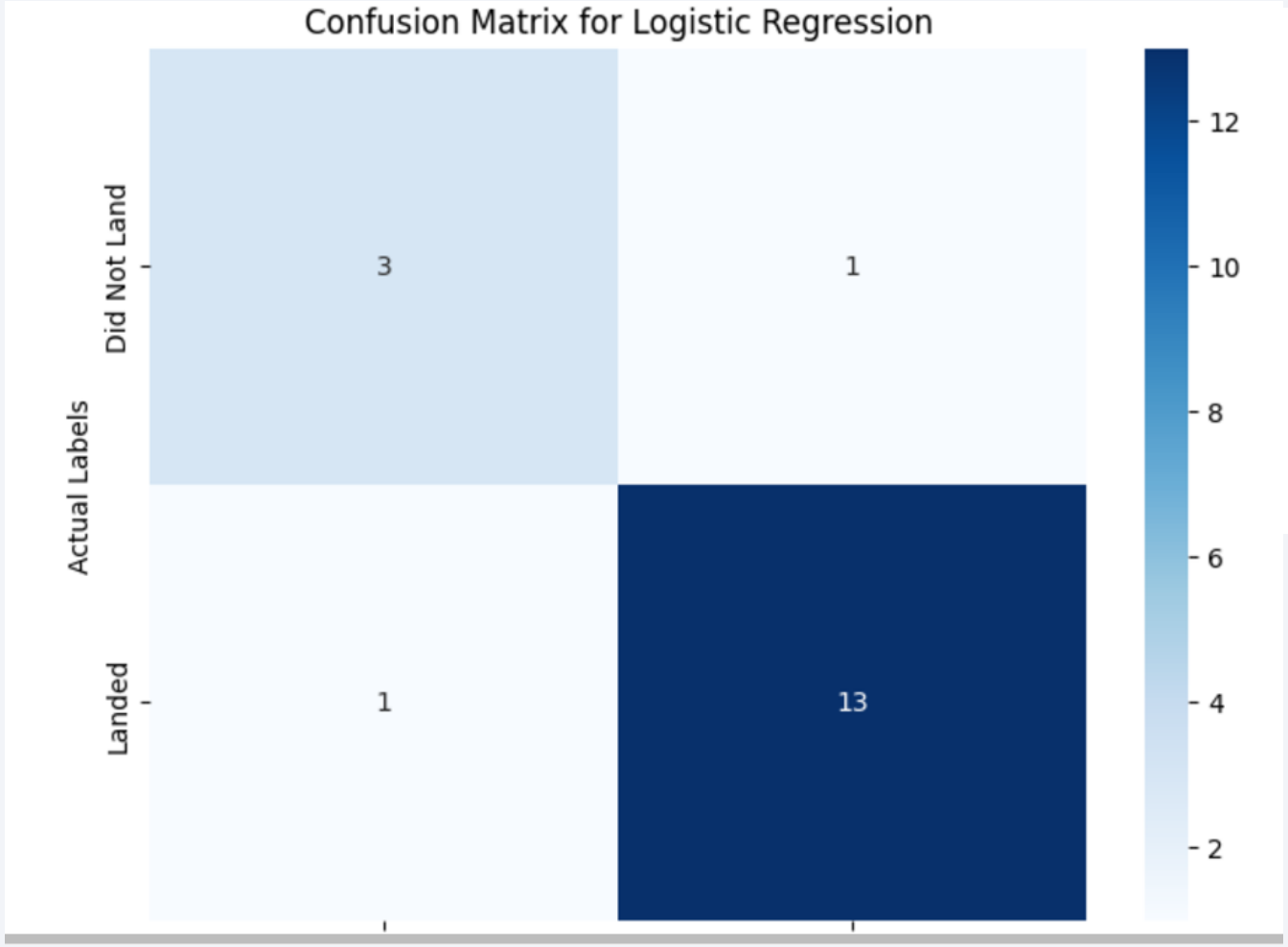
Predictive Analysis (Classification)

Classification Accuracy



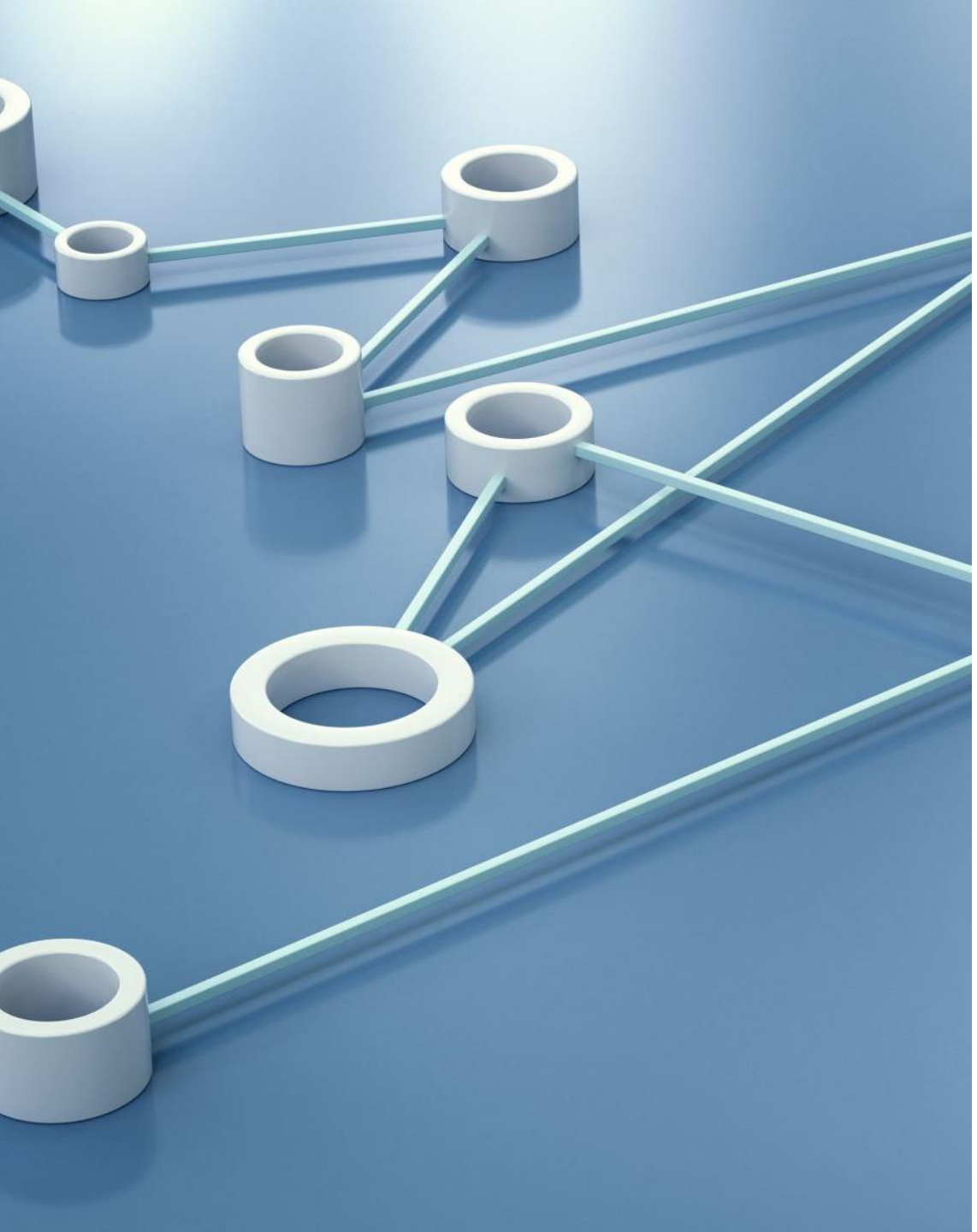
The best model is Support Vector Machine with an accuracy of 0.9444

Confusion Matrix



Best Parameters for Logistic Regression: {'C': 0.1}
Accuracy for Logistic Regression: 0.8889
Classification Report for Logistic Regression:

	precision	recall	f1-score	support
0	0.75	0.75	0.75	4
1	0.93	0.93	0.93	14
accuracy			0.89	18
macro avg	0.84	0.84	0.84	18
weighted avg	0.89	0.89	0.89	18



Conclusions

- In conclusion, this analysis comprehensively explored SpaceX launch data through web scraping, data preprocessing, exploratory data analysis, and interactive visualization using tools like Folium and Seaborn. SQL queries provided insights into payload distribution, launch outcomes, and site-specific trends. Classification models were built and optimized to predict landing success, with SVM achieving the highest accuracy. These methodologies showcased how data-driven approaches can uncover trends, optimize predictions, and enhance decision-making for SpaceX missions. This work underscores the potential of machine learning and data visualization in driving innovation in aerospace and beyond.

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project

Thank you!