IBM Developer
SKILLS NETWORK

# Winning Space Race with Data Science

**Jisahac Olguin**
May 23, 2024

# CONTENTS


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

# Executive Summary

- **Summary of methodologies**
  - Data Collection
  - Data Wrangling
  - Exploratory Data Analysis (EDA)
  - Predictive Modeling:
- **Summary of all results**
  - Launch Site Efficacy
  - Booster Version Impact
  - Interactive maps and dashboard

# Introduction

- **Project background and context**
  - This project leverages comprehensive SpaceX launch data to explore patterns and insights that could enhance the success rates of future launches. SpaceX, a frontrunner in aerospace manufacture and space transport services, aims to reduce space transportation costs and enable the colonization of Mars. Given the complexity and high stakes involved in space launches, it is crucial to analyze historical launch data to identify factors contributing to mission outcomes.
- **Problems you want to find answers**
  - What factors most significantly affect the success of a launch?
  - How does the choice of launch site influence mission success?
  - Are there identifiable trends in payload masses that correlate with successful missions?
  - What impact does the booster version have on launch outcomes?

Section 1

# Methodology

# Methodology

## Executive Summary

- Data collection methodology:
  - Utilized SpaceX REST API
  - Web scraping techniques
- Perform data wrangling
  - Performed data cleaning by removing irrelevant columns
  - Applied One Hot Encoding
- 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

- Describe how data sets were collected.
  - API Integration:
    - Utilized the SpaceX REST API to systematically gather launch data.
  - Web Scraping:
    - Extracted additional launch records from Wikipedia using Python libraries.
- You need to present your data collection process use key phrases and flowcharts
  - Python Libraries: Utilized requests for API calls and BeautifulSoup for parsing HTML from web pages.
  - Data Cleaning: Removed unnecessary columns to streamline the dataset for analysis.

# Data Collection – SpaceX API

1. **Get API Data:** Pull data using an API. Convert Response to JSON File

2. **Make JSON:** Turn the data into JSON format.

```
data = response.json()
data = pd.json_normalize(data)
```

3. **Change Data:** Adjust the data as needed.

```
getLaunchSite(data)
getPayloadData(data)
getCoreData(data)
getBoosterVersion(data)
```

4. **Make Dictionary**: Put the data into a dictionary

```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
```

- Link

5. **Make DataFrame:** Create a DataFrame from the dictionary

```
data = pd.DataFrame({key:pd.Series(value) for key, value in launch_dict.items()})
```

6. **Filter Data:** Select only the needed parts of the DataFrame.

```
data_falcon9 = data[data['BoosterVersion']!='Falcon 1']
```

7. **Save Data:** Save the final data to a file.

```
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

# Data Collection - Scraping

1. **Fetch HTML Data:** Retrieve data from HTML

```
response = requests.get(static_url)
```

2. **Setup BeautifulSoup**: Initialize BeautifulSoup for parsing

```
page_title = soup.title.text
```

3. **Locate Tables:** Identify all tables in the HTML

```
html_tables = soup.find_all('table')
```

4. **Extract Column Names:** Gather the names of the columns

```
for th in first_launch_table.find_all('th')
    name = extract_column_from_header(th)
    if name is not None and len(name) > 0 :
        column_names.append(name)
```

Link

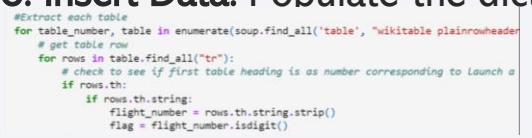
5. **Build Dictionary:** Start a dictionary for data

```
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
del launch_dict['Date and time ( )']

# Let's initial the launch_dict with each value to be an empty list
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

6. **Insert Data:** Populate the dictionary with table data



```
#Extract each table
for table_number, table in enumerate(soup.find_all('table', "wikitable plainrowheader
    # get table row
    for rows in table.find_all("tr"):
        # check to see if first table heading is as number corresponding to launch a
        if rows.th:
            if rows.th.string:
                flight_number = rows.th.string.strip()
                flag = flight_number.isdigit()
```

7. **Make DataFrame:** Convert the dictionary to a DataFrame

```
df= pd.DataFrame({ key:pd.Series(value) for key, value in launch_dict.items() })
```

8. **Save to File**

```
df.to_csv('spacex_web_scraped.csv', index=False)
```

# Data Wrangling

- Let's clarify the outcomes in our dataset regarding rocket landings
  - Failed Landings: Some boosters didn't land successfully.
  - Success Indicators: "True Ocean", "True RTLS", "True ASDS" indicate success.
  - Failure Indicators: "False Ocean", "False RTLS", "False ASDS" indicate failure.
  - Data Conversion: Convert text to categories: 1 for success, 0 for failure.

1. **Count Launches**

2. **Orbit Analysis**

3. **Mission Outcome Tally**: Count outcomes by orbit type

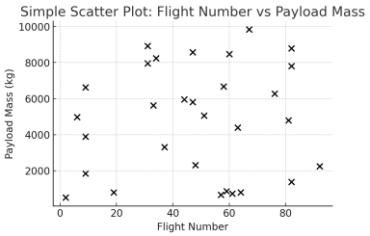
4. **Label Outcomes:** Generate labels from the 'Outcome' column

5. **Save Data:**

Link

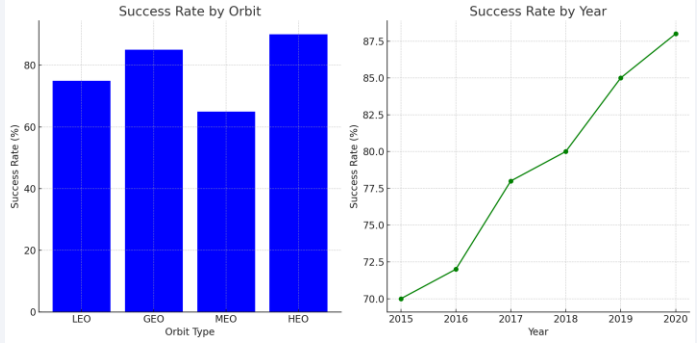
# EDA with Data Visualization

- **Overview of Scatter Plots:**
  - Flight Number against Payload Mass
  - Flight Number against Launch Site
  - Payload against Launch Site
  - Orbit against Flight Number
  - Payload against Orbit Type
  - Orbit against Payload Mass




Link

- **Graph Overview:**
  - **Bar Graph:** Success Rate by Orbit
  - **Line Graph:** Success Rate by Year




- **Explanation:**
  - **Scatter Plots:** Illustrate correlations between variables.
  - **Bar Graphs:** Compare numeric and categorical data.
  - **Line Graphs:** Display trends over time and predict future patterns.

# EDA with SQL

- **We used SQL queries to analyze and retrieve information from the dataset:**
  - Show unique names of launch sites used in the space missions.
  - Show five entries where launch sites start with 'CCA'.
  - Show the total payload mass for boosters launched by NASA (CRS).
  - Show the average payload mass for booster version F9 v1.1.
  - Provide the date of the first successful landing on a ground pad.
  - Provide names of boosters successful in drone ship landings with a payload mass between 4000 and 6000.
  - Count the total successful and failed mission outcomes.
  - List booster versions that have carried the maximum payload.
  - Show records displaying month names, failed drone ship landing outcomes, booster versions, and launch sites for the year 2015.
  - Rank successful landing outcomes from the date 04-06-2010 to 20-03-2017 in descending order.Add the GitHub URL of your completed EDA with SQL notebook, as an external reference and peer-review purpose

Link

# Build an Interactive Map with Folium

- **The Folium map centers on the NASA Johnson Space Center in Houston, Texas:**
  - A red circle marks the NASA Johnson Space Center, labeled with its name using folium.Circle and folium.map.Marker.
  - Red circles pinpoint each launch site, each labeled with the site's name utilizing folium.Circle, folium.map.Marker, and folium.features.DivIcon.
  - Clusters of points represent varied and multiple data for the same coordinates, facilitated by folium.plugins.MarkerCluster.
  - Markers indicate the results of landings: green for successes and red for failures, implemented through folium.map.Marker and folium.Icon.
  - Markers also track the distance from launch sites to crucial infrastructure like railways, highways, coastlines, and cities, drawing lines between them with folium.map.Marker, folium.PolyLine, and folium.features.DivIcon.
- **These elements are designed to enhance understanding of the data and issues, clearly displaying launch sites, their environments, and the outcomes of landings.**

Link

# Build a Dashboard with Plotly Dash

- **The dashboard features components like a dropdown, pie chart, rangeslider, and scatter plot:**

    - The dropdown enables selection of a specific launch site or all sites (dash_core_components.Dropdown).
    - The pie chart displays the count of successful and failed launches for the selected site (plotly.express.pie).
    - The rangeslider facilitates selection within a specified range of payload mass (dash_core_components.RangeSlider).
    - The scatter plot illustrates the correlation between success and payload mass (plotly.express.scatter).Explain why you added those plots and interactions
    - Add the GitHub URL of your completed Plotly Dash lab, as an external reference and peer-review purpose

- **The dashboard includes interactive elements like a dropdown, pie chart, rangeslider, and scatter plot to allow users to select specific launch sites, visualize success and failure rates, refine data by payload mass, and analyze relationships between key variables.**

Link

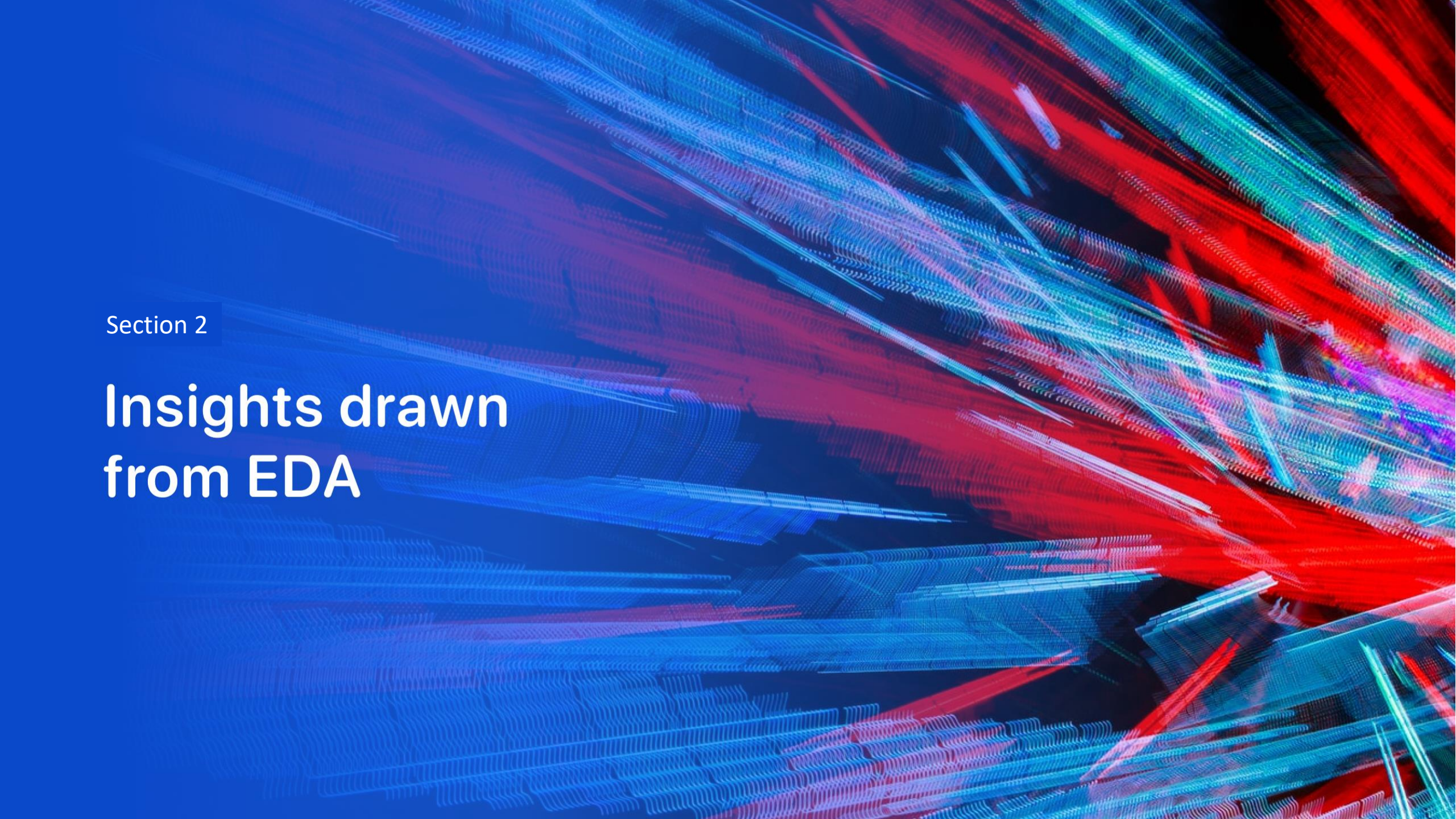
# Predictive Analysis (Classification)

- **Data Setup**
  - Import the dataset.
  - Standardize the data values.
  - Divide the dataset into training and test segments.
- **Model Development**
  - Choose appropriate machine learning techniques.
  - Configure parameters for algorithms using GridSearchCV.
  - Train models using the training data.
- **Model Assessment**
  - Determine the optimal hyperparameters for each model.
  - Evaluate each model's accuracy on the test data.
  - Create a Confusion Matrix to visualize accuracy.
- **Model Comparison**
  - Analyze and compare the accuracy of each model.
  - Select the model that demonstrates the highest accuracy (refer to the Notebook for details).

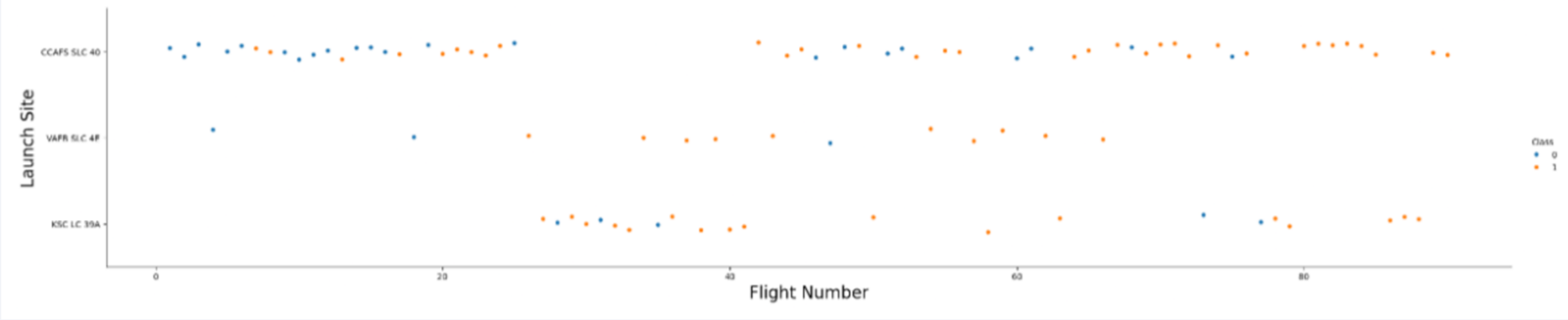
Link

# Results

- **EDA Insights:**
  - Key findings and patterns from data analysis.
- **Model Performance:**
  - Accuracy scores and a snapshot of the Confusion Matrix.
- **Predictive Outcomes:**
  - Highlights of the model's accuracy and example predictions.
- **Model Selection:**
  - Brief comparison and rationale for choosing the top model.

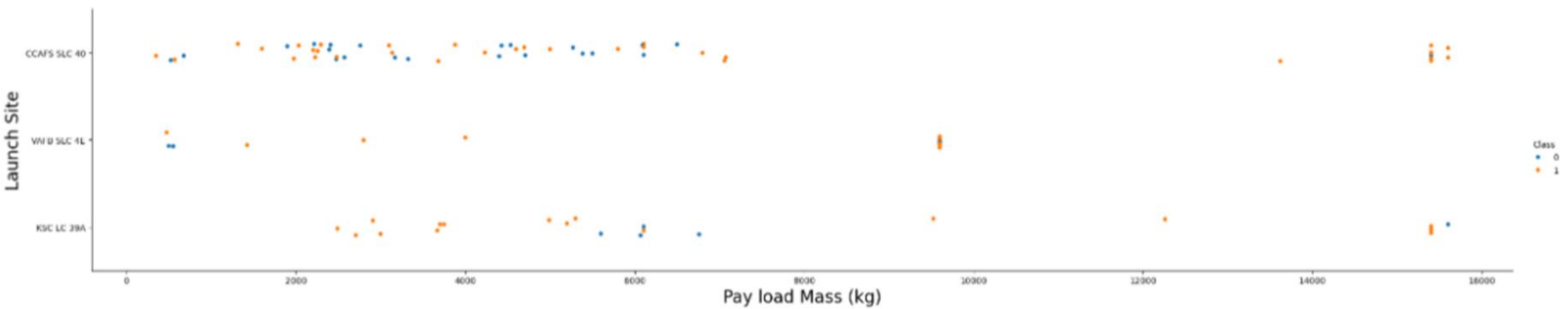

Section 2
Insights drawn from EDA

# Flight Number vs. Launch Site




The graph vividly illustrates a consistent trend of success across different launch sites, highlighting their robust and effective operations over successive flights.

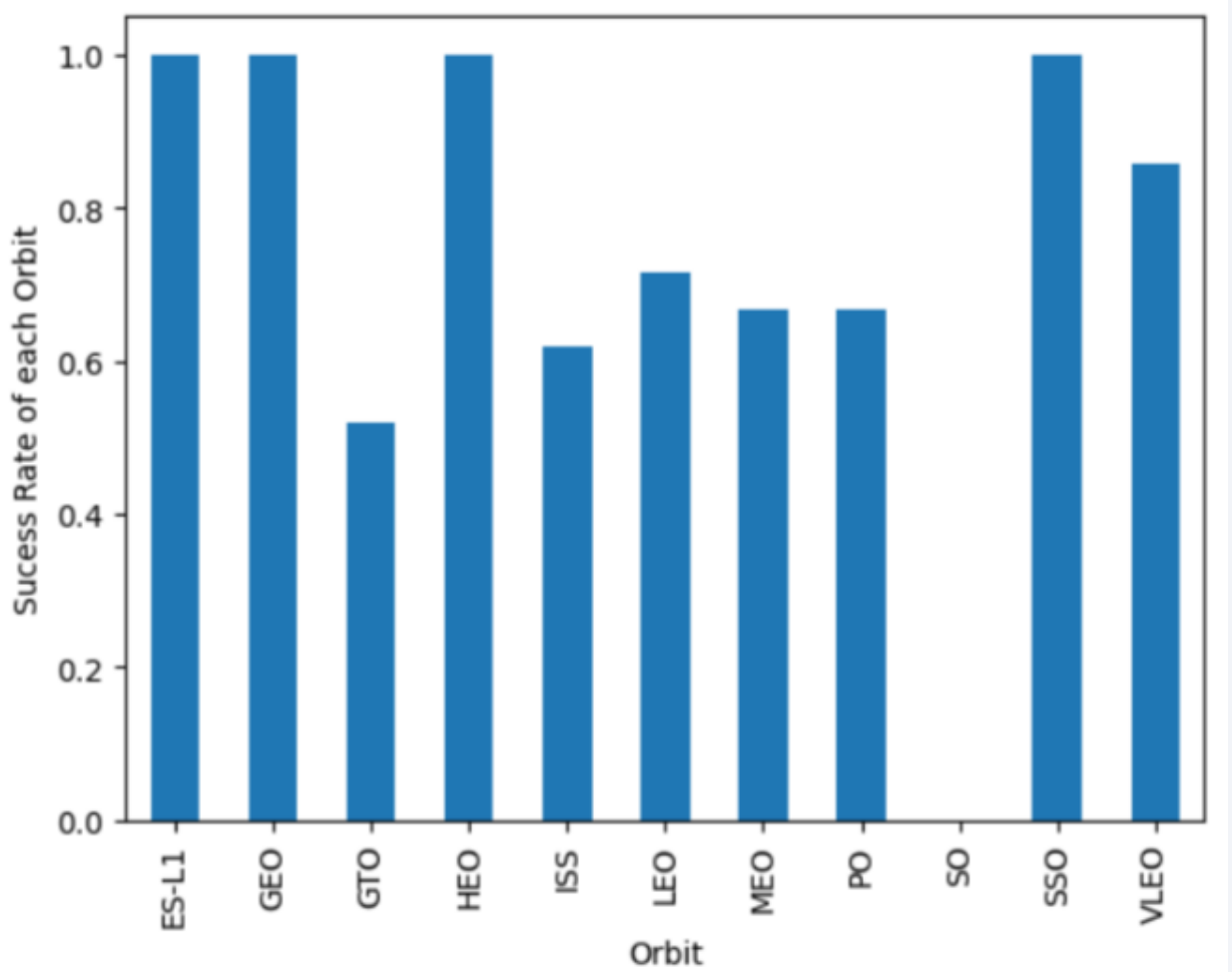
# Payload vs. Launch Site




The success of a landing can depend on the payload weight, with heavier payloads sometimes necessary for success at certain launch sites. However, excessively heavy payloads might compromise the landing's success.

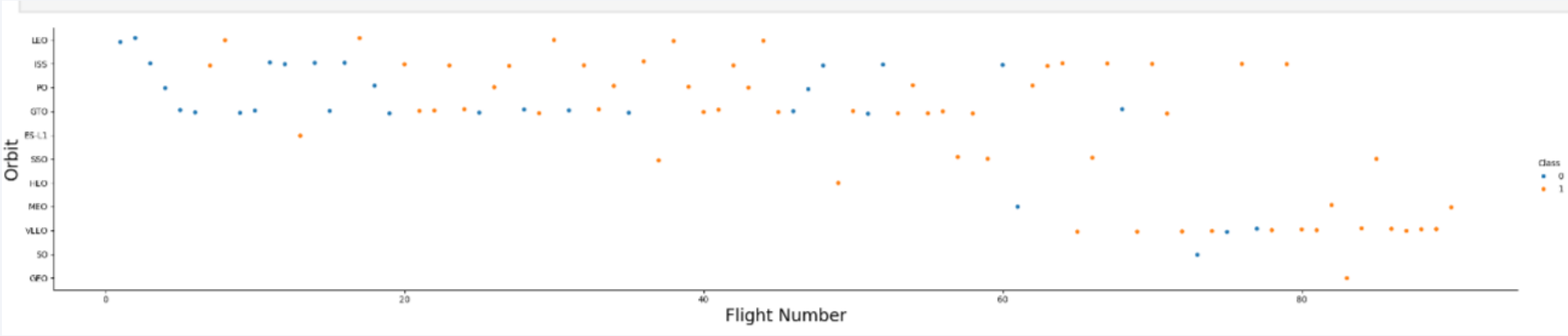
# Success Rate vs. Orbit Type

This chart demonstrates the success rates across various orbit types, revealing that orbits like ES-L1, GEO, HEO, and SSO achieve the highest success rates.

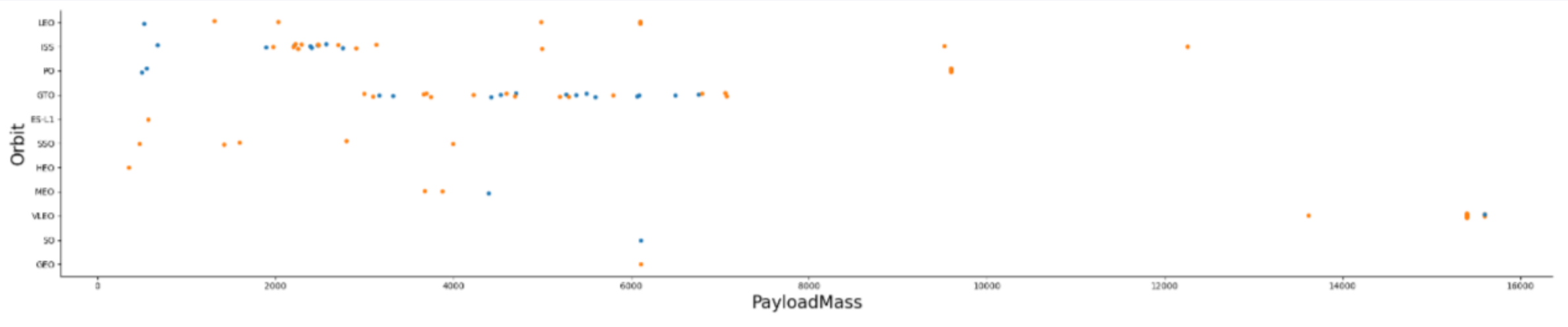


# Flight Number vs. Orbit Type

The data shows that for the LEO orbit, success rate tends to rise as the number of flights increases. Conversely, for orbits like GTO, there doesn't seem to be a direct link between the number of flights and success rates. However, the higher success rates observed in orbits such as SSO or HEO could likely be attributed to the expertise and insights gained from previous launches in other orbits.



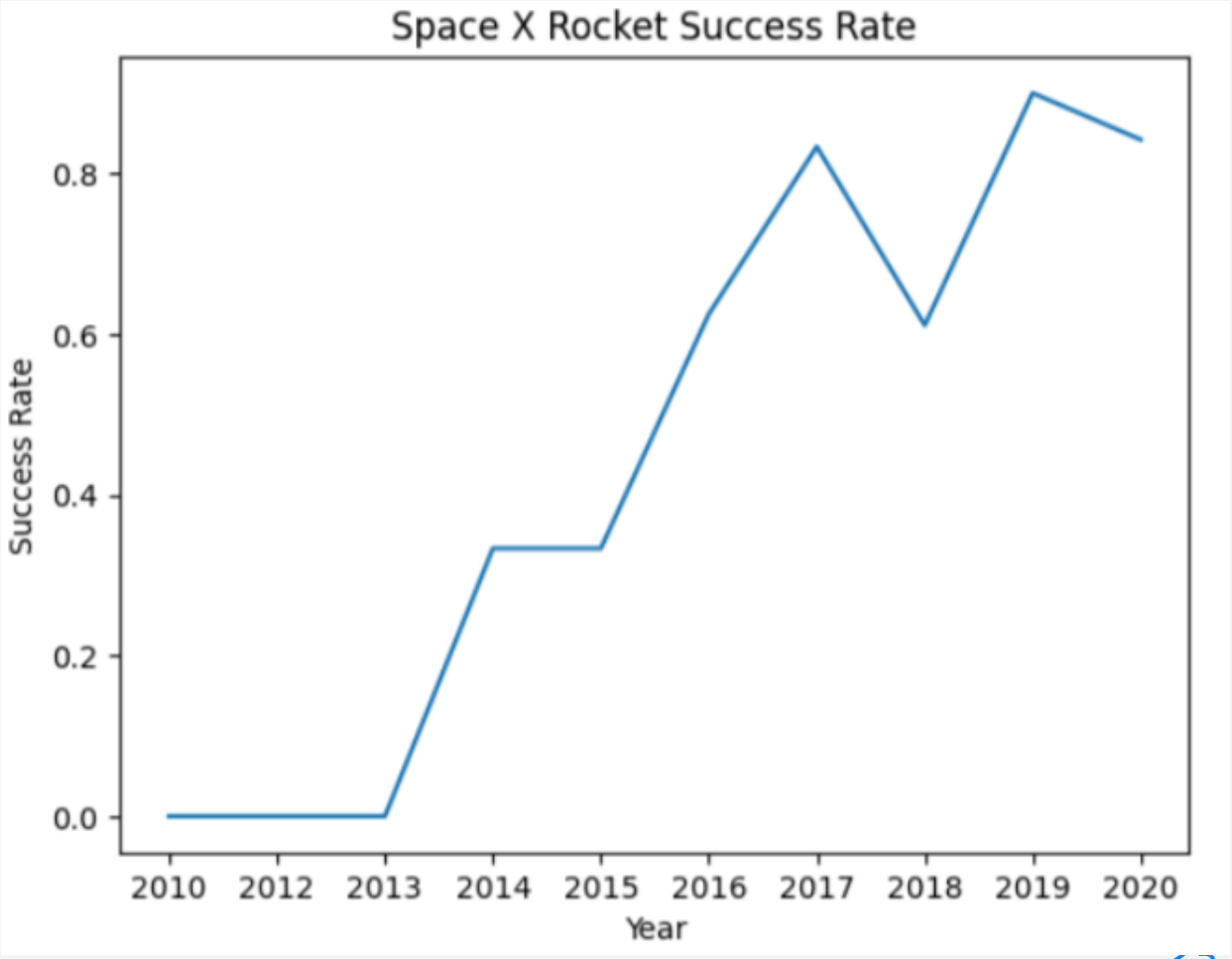
# Payload vs. Orbit Type




Payload weight significantly impacts the success rates of launches within specific orbits. Notably, heavier payloads tend to enhance the success rate in the LEO orbit, while in GTO orbits, reducing the payload weight appears to boost the likelihood of a successful launch.

# Launch Success Yearly Trend

Starting from 2013, there has been a noticeable rise in the success rate of SpaceX rocket launches



# All Launch Site Names

- SQL Query

```
%sql SELECT DISTINCT "LAUNCH_SITE" FROM SPACEXTBL
```

- Results

| Launch_Site |
| --- |
| CCAFS LC-40 |
| VAFB SLC-4E |
| KSC LC-39A |
| CCAFS SLC-40 |

**Explanation:**

Using the DISTINCT keyword in the query effectively eliminates any duplicate entries for LAUNCH_SITE, ensuring each site is counted only once. This provides a clearer and more accurate overview of the data.

# Launch Site Names Begin with 'CCA'

## SQL Query

```
%sql SELECT * FROM SPACEXTBL WHERE "LAUNCH_SITE" LIKE '%CCA%' LIMIT 5
```

## Results

| Date | Time (UTC) | Booster_Version | Launch_Site | Payload | PAYLOAD_MASS__KG_ | Orbit | Customer | Mission_Outcome | Landing_Outcon |
|---|---|---|---|---|---|---|---|---|---|
| :010-<br>)6-04 | 18:45:00 | F9 v1.0 B0003 | CCAFS LC-40 | Dragon Spacecraft Qualification Unit | 0 | LEO | SpaceX | Success | Failure (parachut |
| :010-<br>2-08 | 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 (parachut |
| :012-<br>)5-22 | 7:44:00 | F9 v1.0 B0005 | CCAFS LC-40 | Dragon demo flight C2 | 525 | LEO (ISS) | NASA (COTS) | Success | No attem |
| :012-<br>0-08 | 0:35:00 | F9 v1.0 B0006 | CCAFS LC-40 | SpaceX CRS-1 | 500 | LEO (ISS) | NASA (CRS) | Success | No attem |
| :013-<br>)3-01 | 15:10:00 | F9 v1.0 B0007 | CCAFS LC-40 | SpaceX CRS-2 | 677 | LEO (ISS) | NASA (CRS) | Success | No attem |

## Explanation:

The WHERE clause, combined with the LIKE clause, filters out launch sites containing the substring 'CCA'. The LIMIT 5 command then displays only the first five results from this filtered list, providing a concise snapshot.

# Total Payload Mass

**SQL Query**

```
%sql SELECT SUM("PAYLOAD_MASS__KG_") FROM SPACEXTBL WHERE "CUSTOMER" = 'NASA (CRS)'
```

**Results**

| SUM("PAYLOAD_MASS__KG_") |
|---|
| 45596 |

**Explanation**

The query calculates the total payload mass for missions where NASA (CRS) is listed as the customer, summarizing the cumulative weight carried across these specific launches.

# Average Payload Mass by F9 v1.1

**SQL Query**

```
%sql SELECT AVG("PAYLOAD_MASS__KG_") FROM SPACEXTBL WHERE "BOOSTER_VERSION" LIKE '%F9 v1.1%'
```

**Results**

| AVG("PAYLOAD_MASS__KG_") |
|---|
| 2534.6666666666665 |

**Explanation**

This query determines the average payload mass for launches using booster versions that include 'F9 v1.1' in their designation, offering an overall mean weight for these boosters.

# First Successful Ground Landing Date

### SQL Query

```
%sql SELECT MIN("DATE") FROM SPACEXTBL WHERE "Landing _Outcome" LIKE '%Success%'
```

### Explanation

The WHERE clause narrows the dataset to include only successful landings. Using the MIN function, it then identifies the earliest date from these records.

# Successful Drone Ship Landing with Payload between 4000 and 6000

## SQL Query

```
%sql SELECT "BOOSTER_VERSION" FROM SPACEXTBL WHERE "LANDING _OUTCOME" = 'Success (drone ship)' \
AND "PAYLOAD_MASS__KG_" > 4000 AND "PAYLOAD_MASS__KG_" < 6000;
```

## Explanation:

This query identifies booster versions that successfully landed with payloads between 4000 and 6000 kg.
The WHERE and AND clauses are used to filter the dataset accordingly.

# Total Number of Successful and Failure Mission Outcomes

## SQL Query

```
%sql SELECT (SELECT COUNT("MISSION_OUTCOME") FROM SPACEXTBL WHERE "MISSION_OUTCOME" LIKE '%Success%') AS SUCCESS, \
(SELECT COUNT("MISSION_OUTCOME") FROM SPACEXTBL WHERE "MISSION_OUTCOME" LIKE '%Failure%') AS FAILURE
```

## Results

| SUCCESS | FAILURE |
|---|---|
| 100 | 1 |

## Explanation

The first SELECT displays subqueries with results: one subquery counts successful missions and the other counts unsuccessful ones. The WHERE clause with LIKE filters the mission outcomes, and the COUNT function tallies the filtered records.

# Boosters Carried Maximum Payload

### SQL Query

```
%sql SELECT DISTINCT "BOOSTER_VERSION" FROM SPACEXTBL \
WHERE "PAYLOAD_MASS__KG_" = (SELECT max("PAYLOAD_MASS__KG_") FROM SPACEXTBL)
```

### Results

| 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 |

### Explanation:

A subquery identifies the heaviest payload using the MAX function. The main query then uses this result to return the unique booster versions (SELECT DISTINCT) that carried this heaviest payload.

# 2015 Launch Records

### SQL Query

```
%sql SELECT substr("DATE", 4, 2) AS MONTH, "BOOSTER_VERSION", "LAUNCH_SITE" FROM SPACEXTBL\
WHERE "LANDING _OUTCOME" = 'Failure (drone ship)' and substr("DATE",7,4) = '2015'
```

## Explanation

This query retrieves the month, booster version, and launch site for unsuccessful landings that occurred in 2015. The SUBSTR function extracts the month and year from the date, where SUBSTR(DATE, 4, 2) gives the month and SUBSTR(DATE, 7, 4) gives the year.

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

## SQL Query

```
%sql SELECT "LANDING _OUTCOME", COUNT("LANDING _OUTCOME") FROM SPACEXTBL\
WHERE "DATE" >= '04-06-2010' and "DATE" <= '20-03-2017' and "LANDING _OUTCOME" LIKE '%Success%'\
GROUP BY "LANDING _OUTCOME" \
ORDER BY COUNT("LANDING _OUTCOME") DESC ;
```

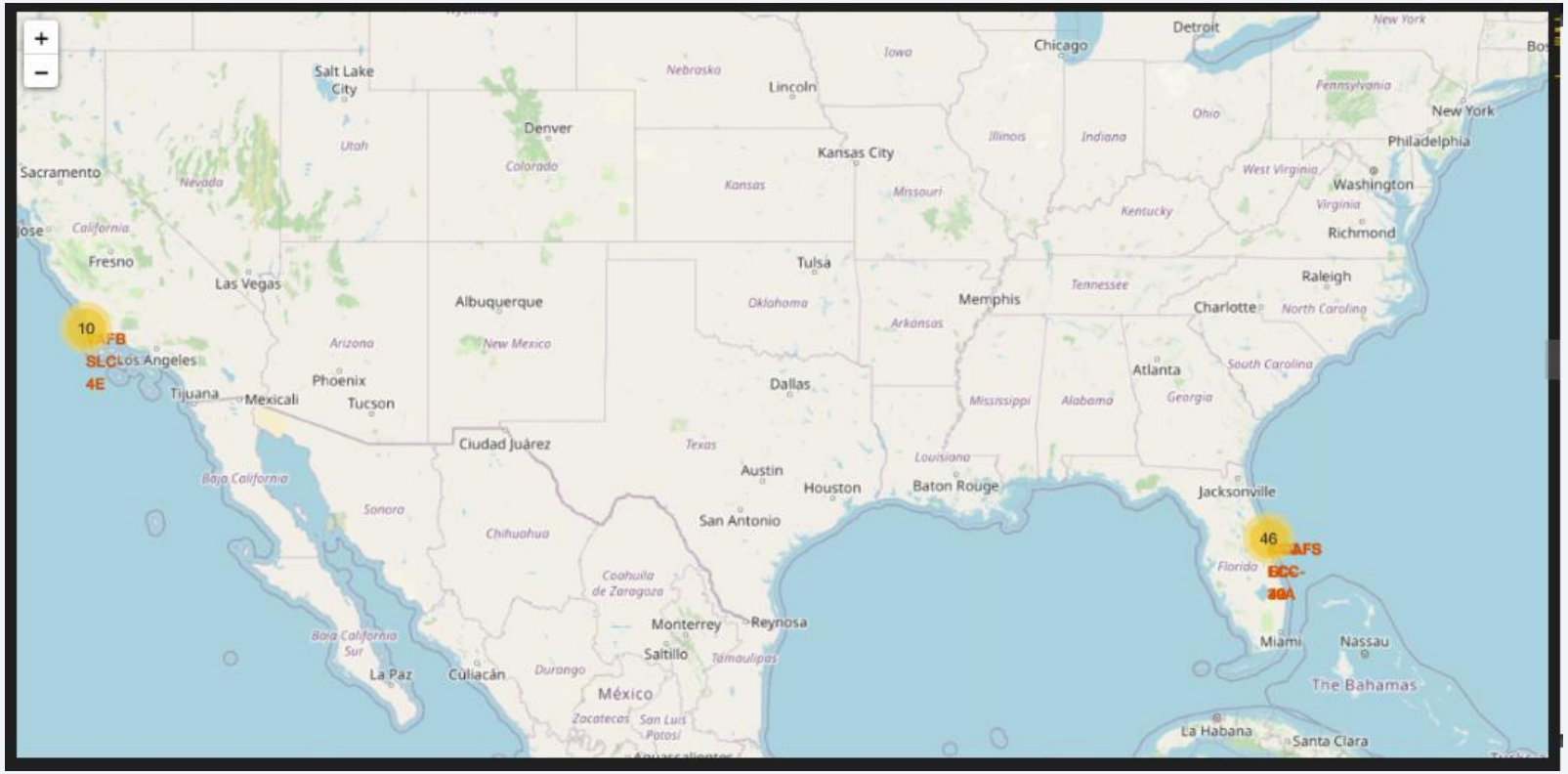
## Explanation

This query retrieves and counts successful landing outcomes for missions between 04/06/2010 and 20/03/2017. The GROUP BY clause organizes the results by landing outcome, and ORDER BY COUNT DESC sorts them in descending order.

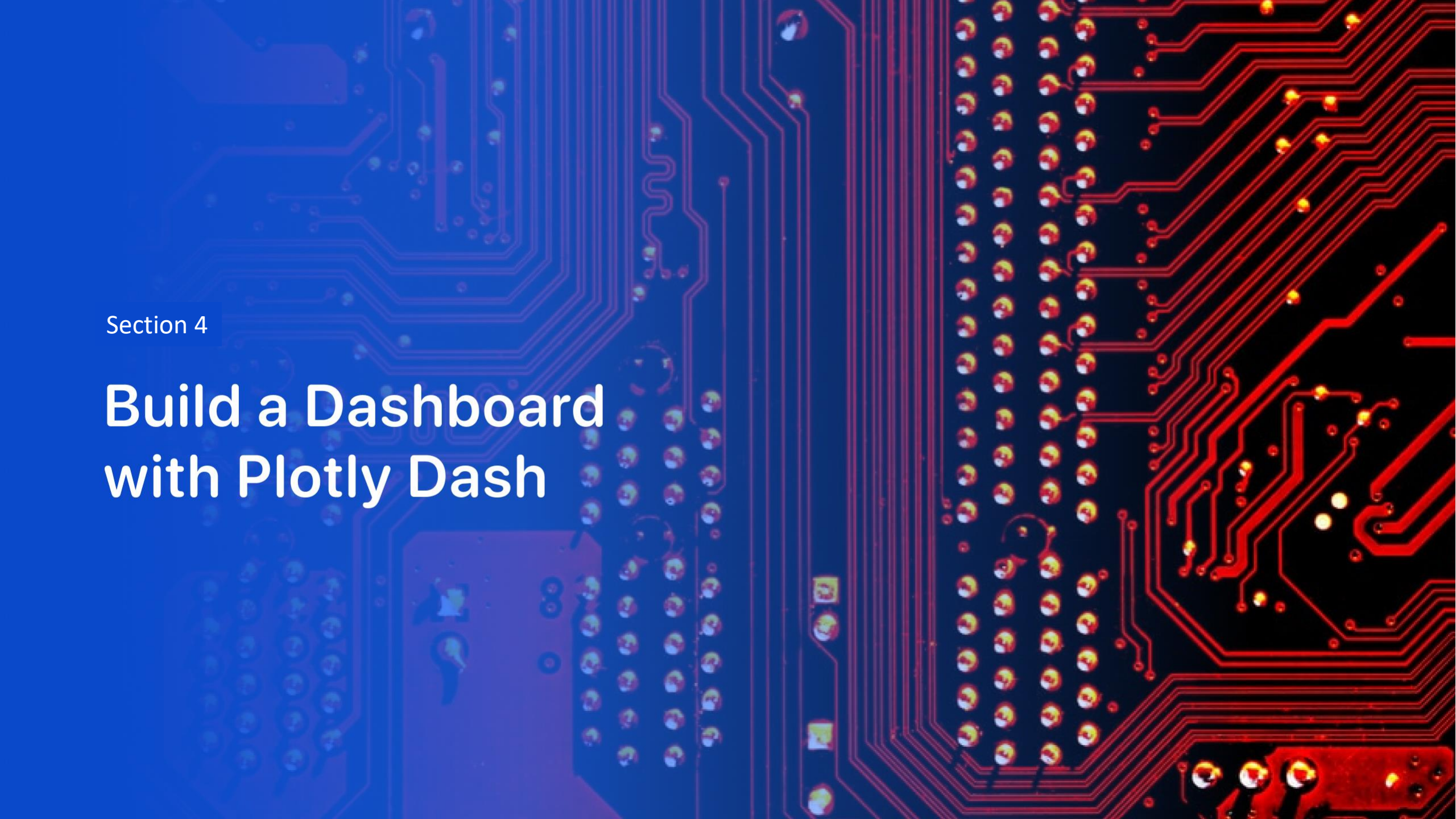
Section 3

# Launch Sites Proximities Analysis

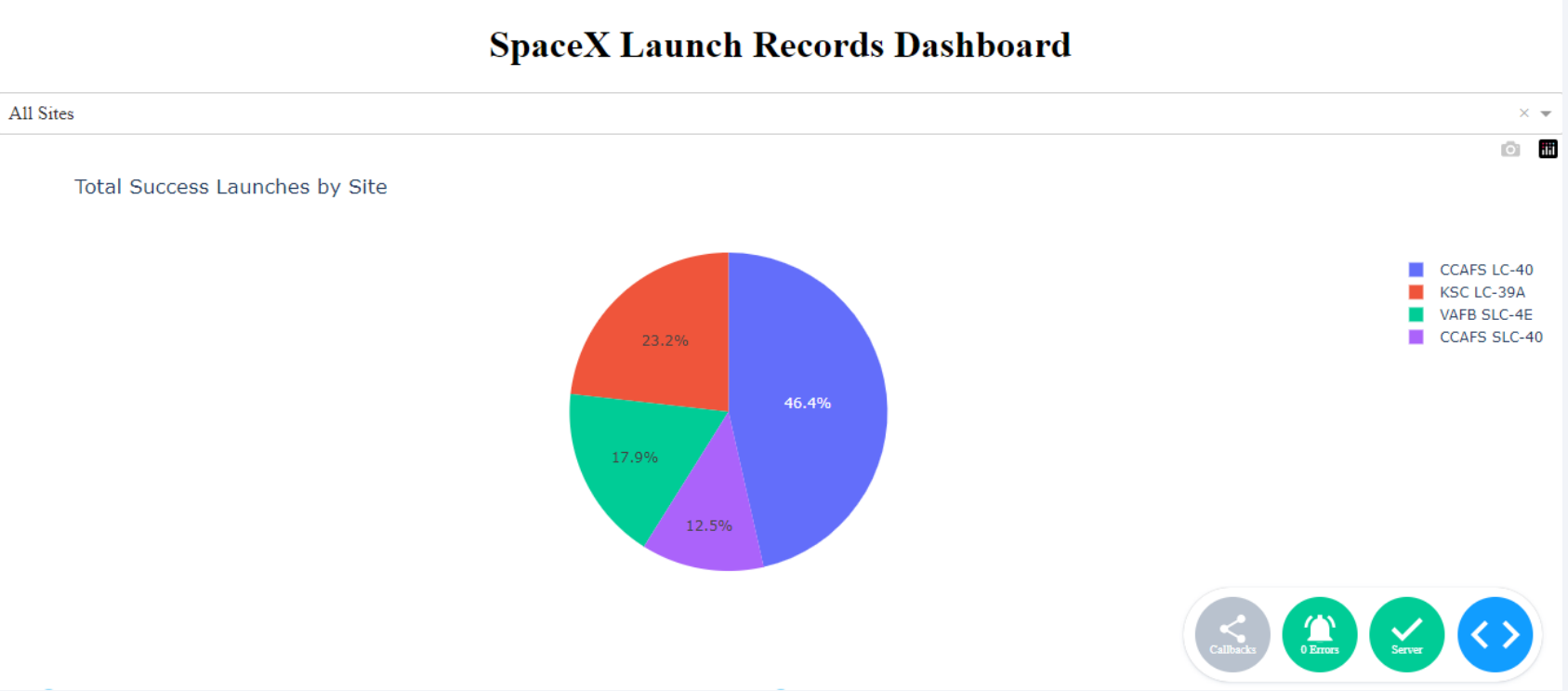
# Ground stations - Folium Map




SpaceX's launch sites are positioned along the coasts of the United States.

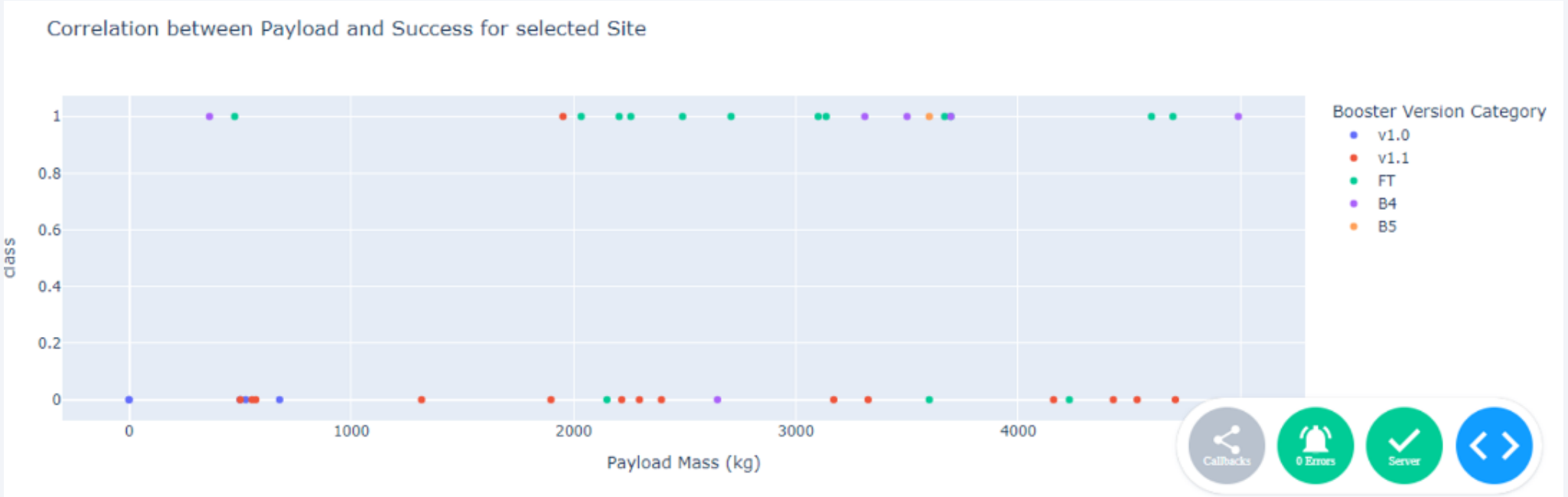

Section 4
Build a Dashboard with Plotly Dash

# <Dashboard Screenshot 1>




CCAFS LC-40: Represented with the largest color segment in the chart, indicating that it has a higher proportion of successful launches.

# <Dashboard Screenshot 3>

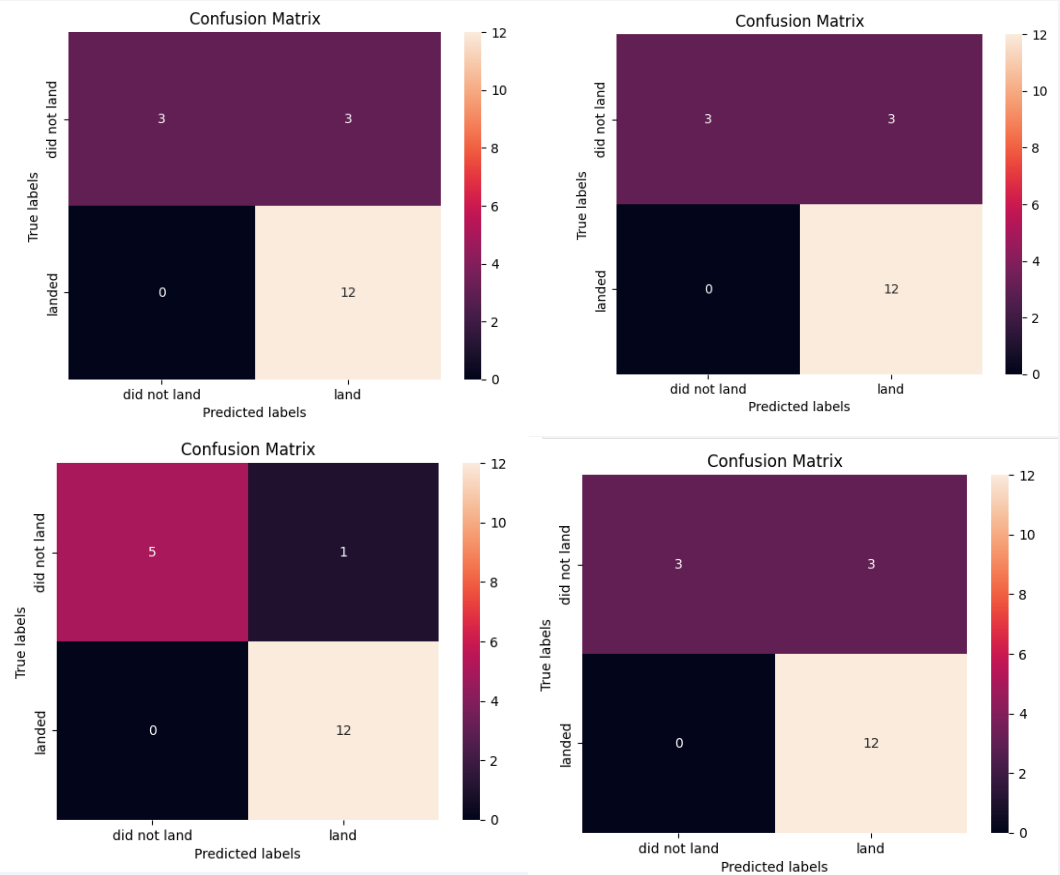



Lighter payloads tend to have higher success rates compared to heavier payloads.

Section 5

# Predictive Analysis (Classification)

# Confusion Matrix













Given that the test accuracies are consistent across three instances, the confusion matrices for these are identical. The primary issue with these models is the occurrence of false positives. However, in one instance, the confusion matrix displays a '5', indicating a variation possibly due to differences in model parameters, data handling, or random seed settings during training.

# Conclusions

- Key Factors for Mission Success:
  - Success is influenced by the launch site, orbit, and experience from previous launches. Repeated launches help improve success rates over time.
- Orbit Success Rates:
  - The most successful orbits are GEO, HEO, SSO, and ES-L1, indicating that specific requirements of these orbits play a crucial role in mission outcomes.
- Payload and Success:
  - Generally, missions with lighter payloads achieve higher success rates, though requirements can vary by orbit.
- Launch Site Performance:
  - CCAFS LC-40 has the highest success rate, suggesting site-specific factors significantly impact outcomes. Additional atmospheric or environmental data could help explain variations between sites.
- Model Selection:
  - Despite similar test accuracies, the Decision Tree Algorithm was chosen due to its better performance in training, highlighting its reliability for predictio

# Thank you!