Winning Space Race with Data Science

John Pitts
October 2023

Outline

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

Executive Summary

Methodological Overview

- API-Based Data Acquisition
- Web Scraping for Data Collection
- Data Refinement and Cleaning
- In-depth Data Examination using SQL
- Visual Data Analysis Exploration
- Folium-Driven Interactive Data Visualization
- Predictive Analysis through Machine Learning

Comprehensive Results Breakdown

- Results from In-depth Data Exploration
- Visual Data Insights via Screenshots
- Outcomes from Predictive Analysis

Introduction

Project Overview and Context

Space X offers Falcon 9 rocket launches at a price point of 62 million dollars, a significant reduction compared to other providers who charge upwards of 165 million dollars. A major factor contributing to this cost-efficiency is Space X's ability to reuse the first stage of the rocket. By predicting the successful landing of this first stage, we can better estimate the overall launch cost. This insight is crucial for any competing company considering a bid against Space X in the rocket launch industry. The primary objective of this project is to develop a machine learning model to forecast the success of the first stage landing.

Key Questions to Address

- Which elements influence the successful landing of the rocket?
- How do different factors interplay to impact the likelihood of a successful landing?
- What are the essential operational conditions required to guarantee a successful landing procedure?

Section 1

Methodology

Methodology

Data Collection

- ❑ Sourced from SpaceX API and web-scraped Wikipedia.

Data Wrangling

- ❑ Applied one-hot encoding to categorical variables.

Exploratory Data Analysis (EDA)

- ❑ Used visualization tools and SQL for data insights.

Interactive Visualization

- ❑ Utilized Folium and Plotly Dash for dynamic visuals.

Predictive Analysis

- ❑ Built, optimized, and assessed classification models.

Data Collection

Summary

- ✓ Initiated a get request to the SpaceX REST API.
- ✓ Decoded the API response into a Json format.
- ✓ Transformed this Json content into a structured pandas data frame.
- ✓ Web scraped launch details from Wikipedia using Beautiful Soup.
- ✓ Extracted launch records from Wikipedia's HTML tables.
- ✓ Integrated the scraped data into the primary pandas dataframe.
- ✓ Conducted a thorough data cleaning process across all sources.
- ✓ Addressed and filled any missing values to ensure data completeness.

Data Collection – SpaceX API

I collected data from the SpaceX API using a GET request, then cleaned and formatted the retrieved data through basic wrangling techniques.

Below is the GitHub URL for my Notebook:

[SpaceX API Link]

Get Data

```
[20]: spacex_url="https://api.spacexdata.com/v4/launches/past"

[23]: response = requests.get(spacex_url)
```

Normalize Data

```
[14]: # Use json_normalize meethod to convert the json result into a dataframe
      import requests
      import pandas as pd
      response = requests.get('https://cf-courses-data.s3.us.cloud-object-storage.appdomain.clo
      data_json = response.json()
      data = pd.json_normalize(data_json)
      print(data.head())
```

Filter only Falcon 9

```
[37]: # Hint data['BoosterVersion']!='Falcon 1'
      data_falcon9 = launch_df[launch_df['BoosterVersion'] == 'Falcon 9']
```

Replace Missing Data

```
[44]: import numpy as np
      mean_payload_mass = data_falcon9['PayloadMass'].mean()
      data_falcon9['PayloadMass'].fillna(mean_payload_mass, inplace=True)
```

Data Collection - Scraping

- I utilized web scraping techniques to extract Falcon 9 launch records using BeautifulSoup. I then transformed the table into a pandas data frame and saved the results to a CSV file.

- Below is the GitHub URL for my Notebook:

[SpaceX Web scrape Link]

Get Data

```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=102
response = requests.get(static_url)
if response.status_code == 200:
    html_content = response.text
    print("HTTP GET request was successful.")
else:
    print(f"HTTP GET request failed with status code: {response.status_code}")
```

```
HTTP GET request was successful.
```

Create Beautiful Soup Object

```
import requests
from bs4 import BeautifulSoup
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldi
response = requests.get(static_url)
if response.status_code == 200:
    html_content = response.text
    soup = BeautifulSoup(html_content, 'html.parser')
    print("BeautifulSoup object created successfully.")
else:
    print(f"HTTP GET request failed with status code: {response.status_code}")
```

```
BeautifulSoup object created successfully.
```

Extract Column info

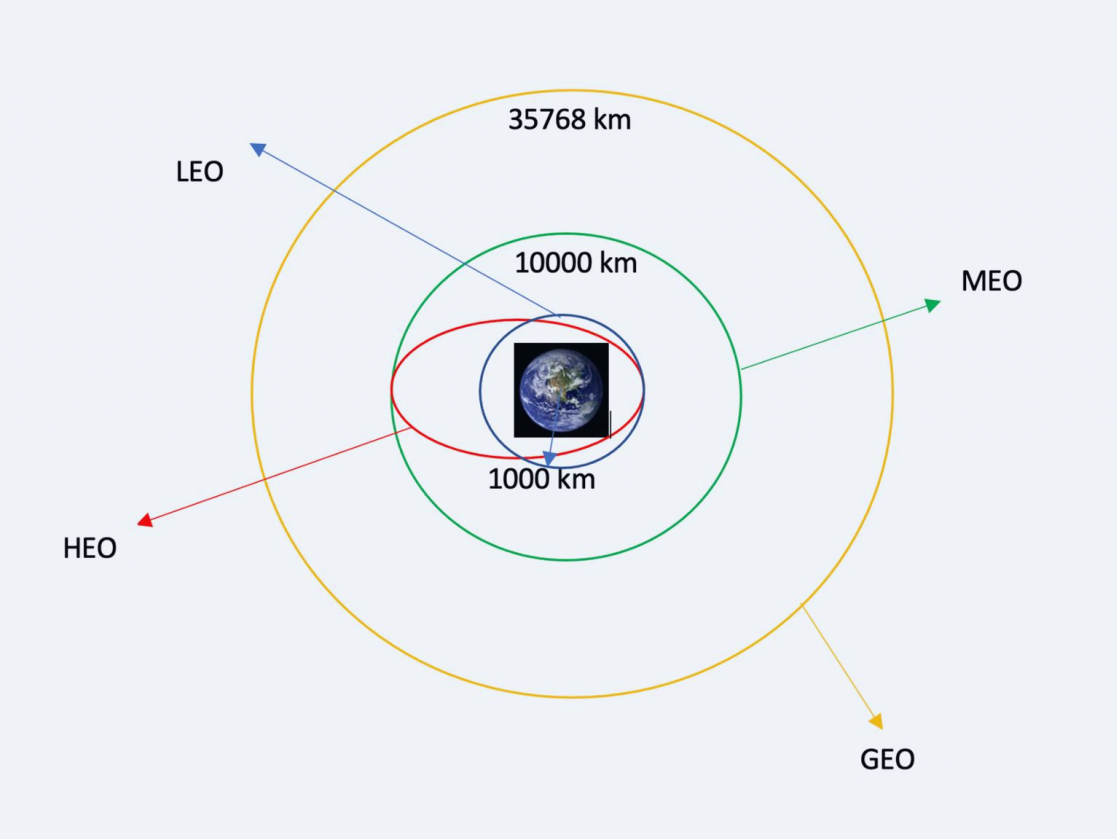
Parse Tables

Create data frame and save file

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

[21]: df.to_csv('spacex_web_scraped.csv', index=False)
```

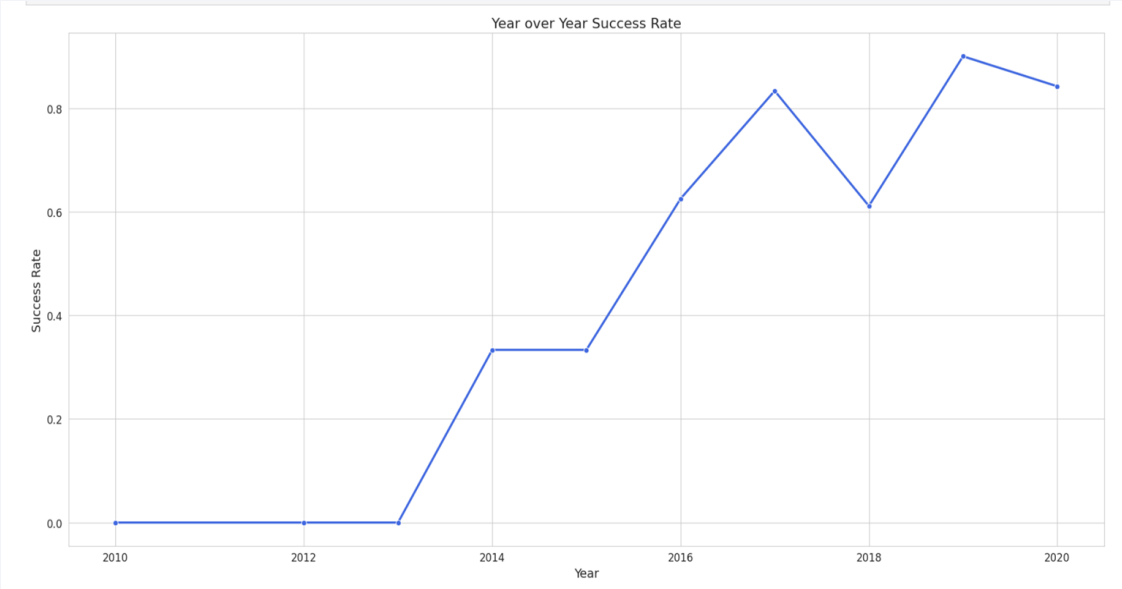
Data Wrangling

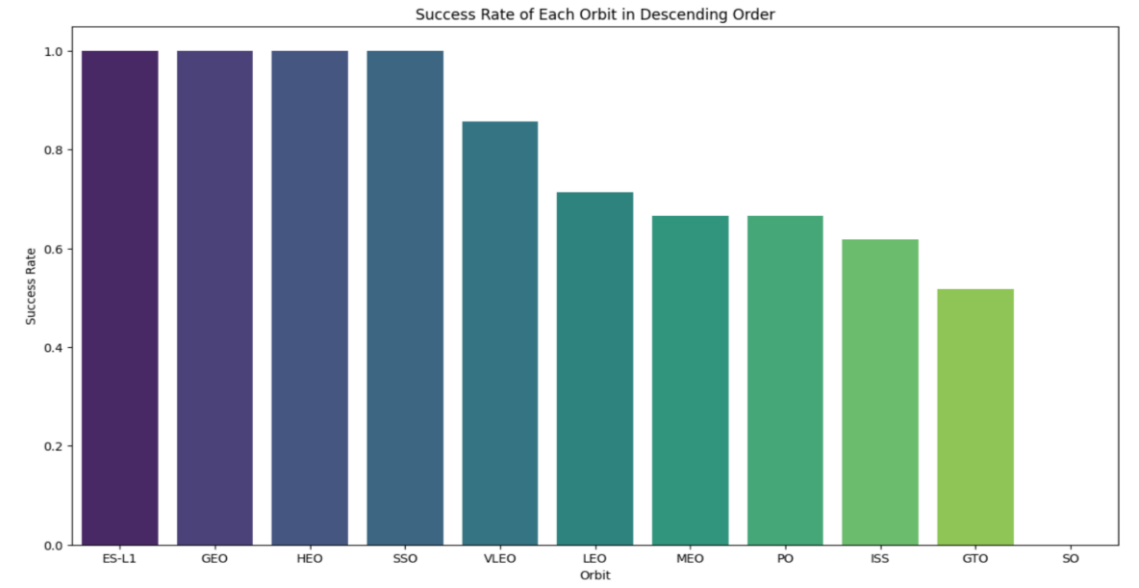
- Analyzed data to determine insightful patterns.
- Identified launch variety by site and orbit.
- Simplified mission outcomes into 'success' or 'failure'.
- Categorized landings, using terms like 'True Ocean' and 'True RTLS'.
- Exported refined data to a .csv file.



[Data Wrangling Notebook Link]

EDA with Data Visualization





Success Line Graph Tracked annual landing success rates, providing a view of mission outcomes over time

Orbit Success Bar Graph: Highlights landing success rates by orbit, revealing the correlation between orbit type and success.

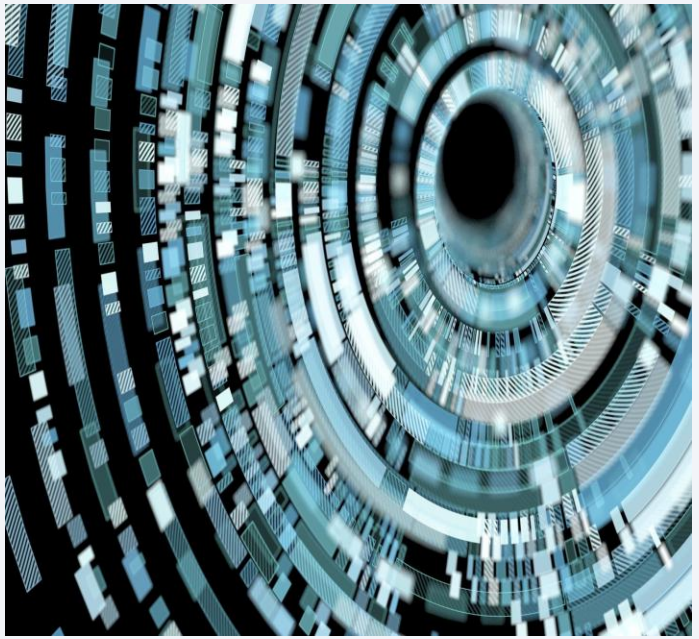
Scatter Plots were also used to show relationships and correlations between variables like Flight Number, Payload Mass, Launch Site, and Orbit.

[Data Visualization Notebook Link]

EDA with SQL

SQL Queries Performed

- Identify unique launch sites in missions.
- Retrieve 5 records of launch sites starting with 'CCA'.
- Total payload mass for NASA's CRS boosters.
- Average payload mass of F9 v1.1 boosters.
- Date of the first successful ground pad landing.
- Boosters with successful drone ship landings and payload between 4000-6000.
- Count of successful vs. failure mission outcomes.
- Booster versions with the maximum payload
- Details for drone ship landing failures in 2015
- Rank landing outcomes between 2010-06-04 and 2017-03-20

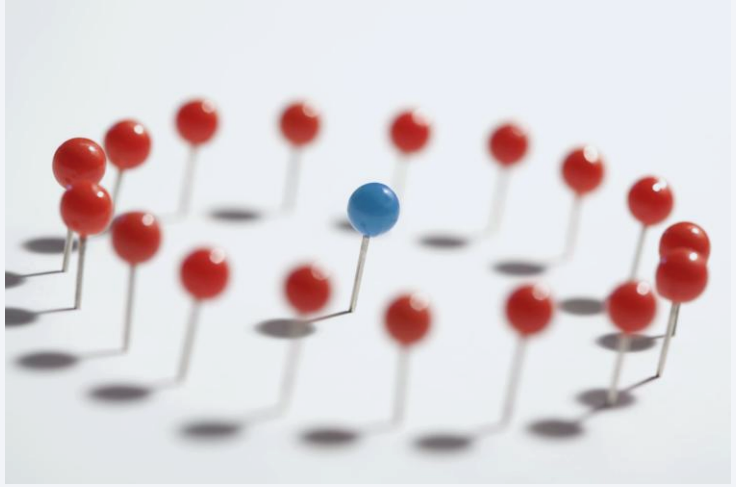


SQL Notebook Link

Build an Interactive Map with Folium

- Visualized Launch Data on an interactive map using the latitude and longitude of each launch site, highlighted by circle markers labeled with the site name.
- Launch outcomes, represented by successes (1) and failures (0), were mapped using green and red markers within a MarkerCluster.
- Distances from launch sites to landmarks were computed to identify patterns. Lines on the map represent these distances.
- Trends observed:
  - Launch sites aren't near railways or highways.
  - Launch sites are close to coastlines.
  - Launch sites maintain a distance from cities.

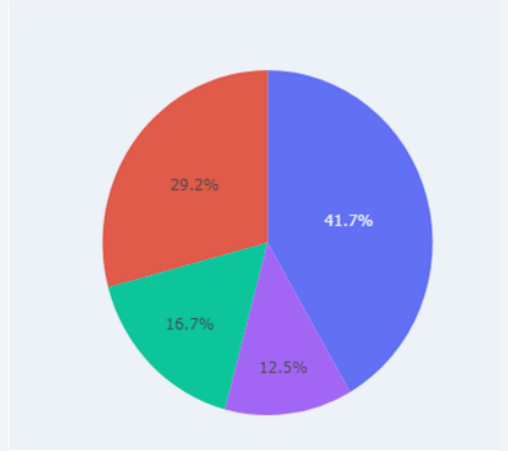
Launch Site Locations Notebook Link



Build a Dashboard with Plotly Dash

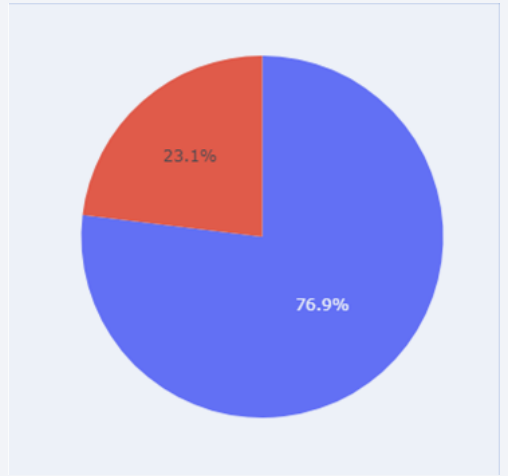
Developed an interactive Plotly dashboard that features:

- ✓ A dropdown menu to select launch sites.
- ✓ Pie charts showcasing the launch success rate.
- ✓ A scatter chart showing launch site, payload mass and outcome.
- ✓ A range slider to filter by payload mass (kg).



Through this dashboard, users can analyze:

- ✓ Which site has the most successful launches.
- ✓ The site with the top success rate.
- ✓ Payload ranges with the best and worst success rates.
- ✓ The F9 Booster version with the highest success rate.



Dashboard Notebook Link

Predictive Analysis (Classification)

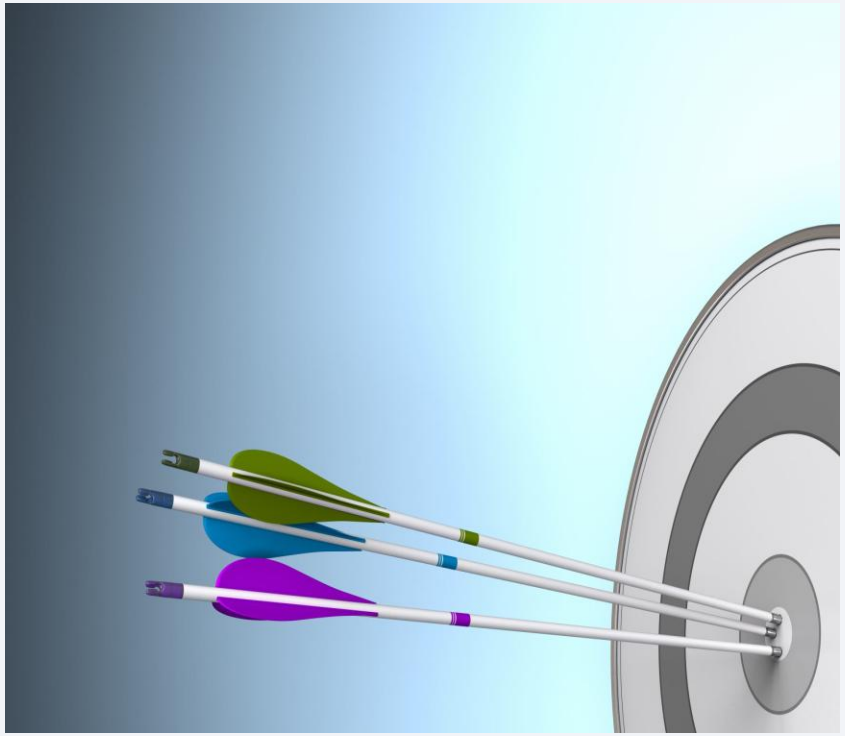
Building Model	Evaluating Model	Improving Model	Finding the Best Performing Classification Model

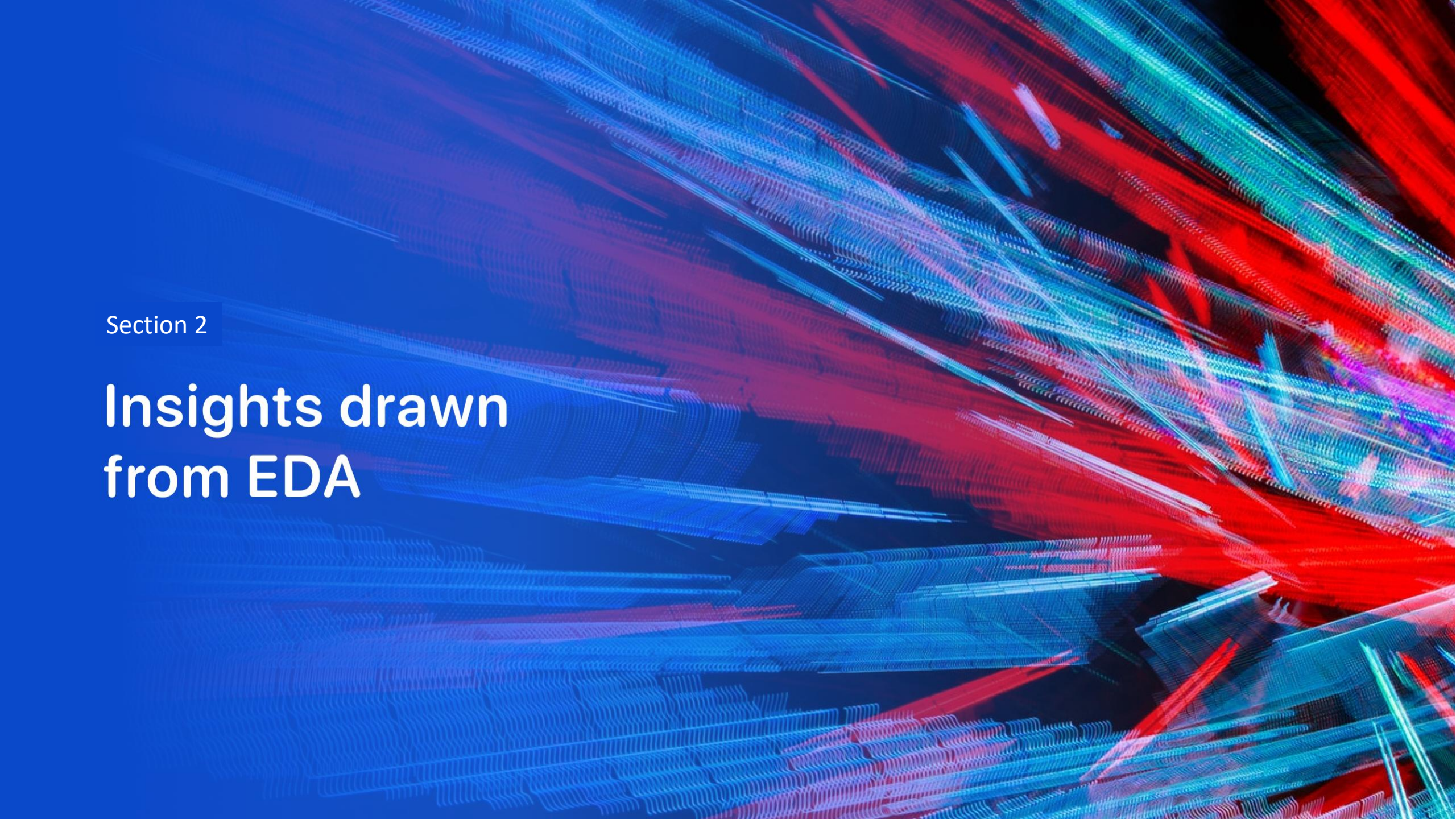
• Load data into NumPy and Pandas.
• Transform and scale features.
• Split into training and test sets.
• Choose machine learning algorithms.
• Apply Grid Search for optimization.
• Train the model.
	• Performance Review:
	• Measure model accuracy.
	• Extract and analyze the tuned hyperparameters.
	• Visualize results, such as with a Confusion Matrix.
		• Feature Refinement:
		  • Engineer or select relevant features to enhance model performance.
		• Algorithm Enhancement:
		  • Fine-tune algorithms or consider alternative models.
			• Result Analysis:
			  • Compare accuracy scores across models.
			  • Determine the top-performing model based on metrics and real-world applicability.

Machine Learning Notebook Link

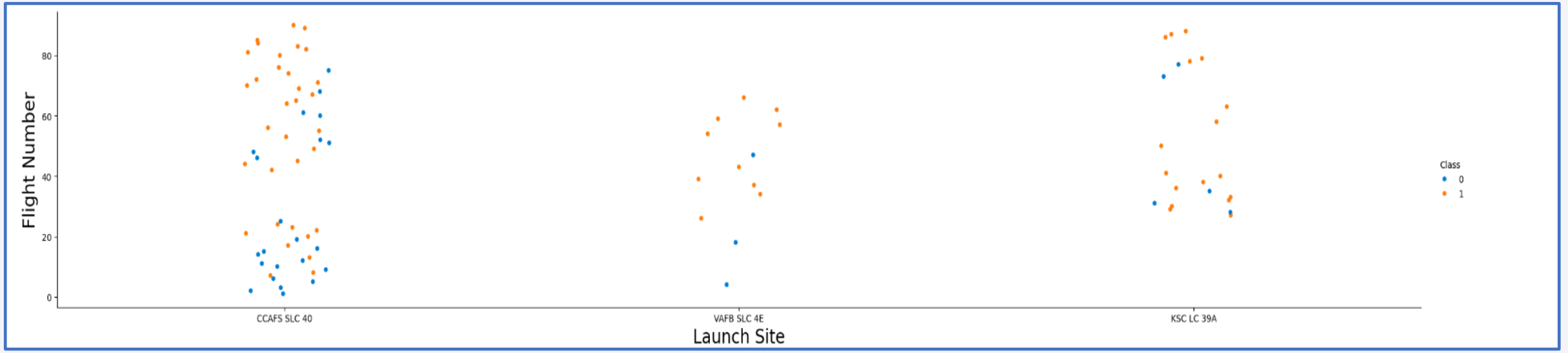
Results

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



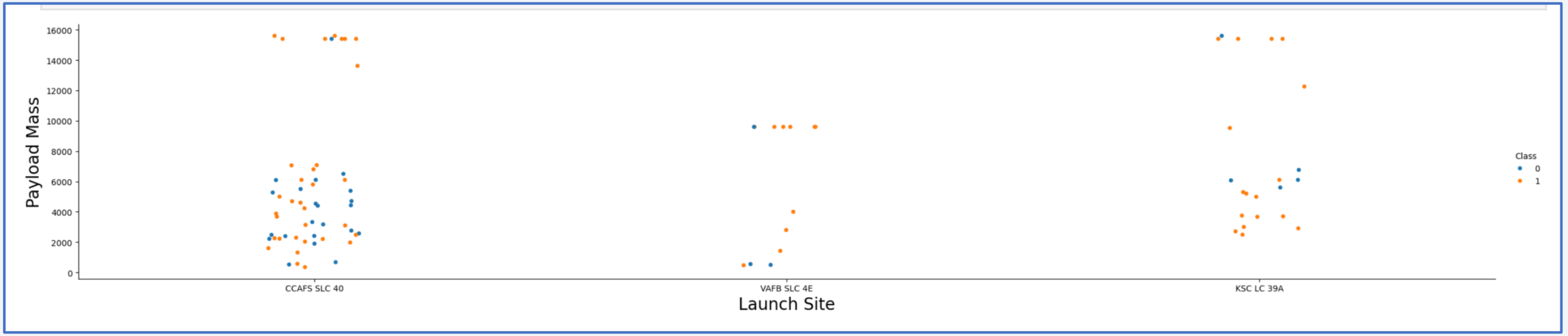

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site



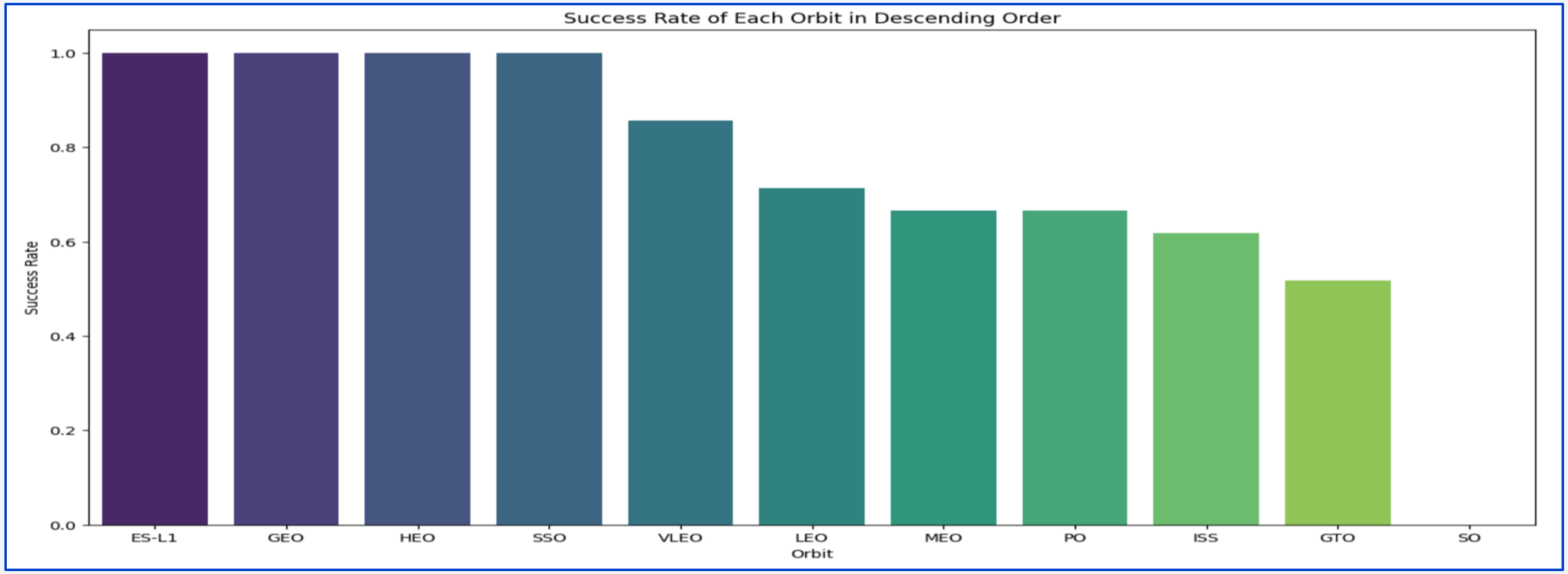
- As flight numbers rise, landings become more successful
- More flights at a launch site correlate with increased landing successes

Payload vs. Launch Site



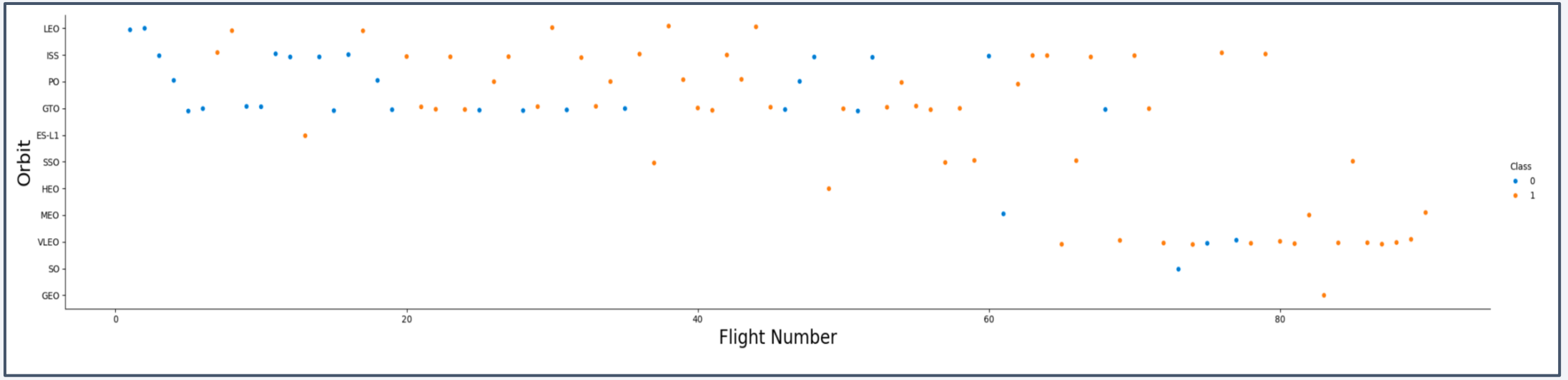
As payload mass at Site CCAFS SLC 40 grows, the likelihood of a successful landing also rises.

Success Rate vs. Orbit Type



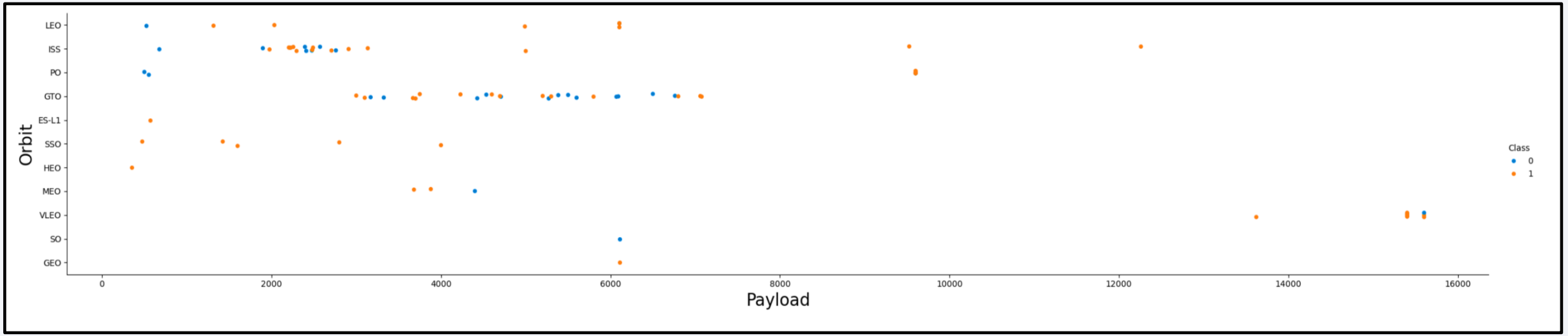
ES-L1, GEO, HEO & SSO have the highest success rates

Flight Number vs. Orbit Type



- There appears to be a correlation between the number of flights and success for LEO and PO orbits
- A correlation between the other orbits and the number of flights is not evident

Payload vs. Orbit Type



- There appears to be a correlation between success and heavy payloads for LEO and ISS orbits
- A correlation between the other orbits and payload mass is not evident

Launch Success Yearly Trend

- The Launch Success Rate rose continually between 2013 and 2020.
- There was a slight dip in the Success Rate in 2018

All Launch Site Names

```
%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

- I used the `SELECT DISTINCT` SQL command to extract all the unique launch sites from the 'SPACEXTABLE'. This ensures that each launch site is listed only once

Launch Site Names Begin with 'CCA'

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

- I used the `SELECT * FROM SPACEXTABLE WHERE Launch_Site LIKE 'CCA%'` query to get the first five entries in 'SPACEXTABLE' where the launch site starts with 'CCA'.

Total Payload Mass

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

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

SUM(PAYLOAD_MASS__KG_)
45596

- I calculate the total payload mass for 'NASA (CRS)' missions in the 'SPACEXTABLE' using `SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTABLE WHERE Customer = 'NASA (CRS)'`

Average Payload Mass by F9 v1.1

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

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

AVG(PAYLOAD_MASS__KG_)
2928.4

- I determine the average payload mass for the 'F9 v1.1' booster version in the 'SPACEXTABLE' using the `SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTABLE WHERE Booster_Version = 'F9 v1.1'

First Successful Ground Landing Date

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

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

MIN(Date)
2015-12-22

- I retrieve the earliest date from the 'SPACEXTABLE' where the landing outcome was a success on a ground pad using the query `SELECT MIN(Date) FROM SPACEXTABLE WHERE Landing_Outcome = 'Success (ground pad)'`. This tells us when the first successful ground pad landing occurred.

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
```

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

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

- I'm using a SQL query to filter out booster versions from the 'SPACEXTABLE' that have a 'Success (drone ship)' landing outcome and a payload mass between 4000 and 6000 kg.

Total Number of Successful and Failure Mission Outcomes

```
%sql SELECT \
    (SELECT COUNT(*) FROM SPACEXTABLE WHERE Mission_Outcome LIKE '%Success%') as Total_Success_Outcomes, \
    (SELECT COUNT(*) FROM SPACEXTABLE WHERE Mission_Outcome LIKE '%Failure%') as Total_Failure_Outcomes;
```

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

Total_Success_Outcomes	Total_Failure_Outcomes
100	1

- Using the `SELECT` command, I'm querying the 'SPACEXTABLE' by applying `SELECT` subqueries. For each subquery, I'm counting rows based on specific criteria within the 'Mission_Outcome' column: one counts the rows with 'Success' and the other counts rows with 'Failure'.

Boosters Carried Maximum Payload

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

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

Booster_Version
F9 B5 B1048.4
F9 B5 B1049.4
F9 B5 B1051.3
F9 B5 B1056.4
F9 B5 B1048.5
F9 B5 B1051.4
F9 B5 B1049.5
F9 B5 B1060.2
F9 B5 B1058.3
F9 B5 B1051.6
F9 B5 B1060.3
F9 B5 B1049.7

- Using the `Select` command, I'm querying the 'SPACEXTABLE' to retrieve the 'Booster_Version'. Specifically, I'm selecting the booster version that has carried the maximum payload mass. This is determined by a subquery that identifies the highest 'PAYLOAD_MASS__KG_' value from the same table.

2015 Launch Records

```
%sql SELECT strftime('%m', Date) as Month, Landing_Outcome, Booster_Version, Launch_Site FROM SPACEXTABLE WHERE Date LIKE '2015-
```

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

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

- Using the `SELECT` command, I'm extracting data from the 'SPACEXTABLE'. I'm retrieving the month of the launch date, landing outcome, booster version, and launch site. Specifically, I'm focusing on records from the year 2015, where the landing outcome indicates a failure and is associated with a drone ship. The `strftime` function is employed to extract the month from the 'Date' column.

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

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

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

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

- Utilizing the `SELECT` command in `%sql`, I'm retrieving the count of each distinct landing outcome from the 'SPACEXTABLE'. The data is limited to a specific date range, from June 4, 2010, to March 20, 2017. The outcomes are then grouped by type and ordered in descending order based on their counts.

Section 3

Launch Sites Proximities Analysis

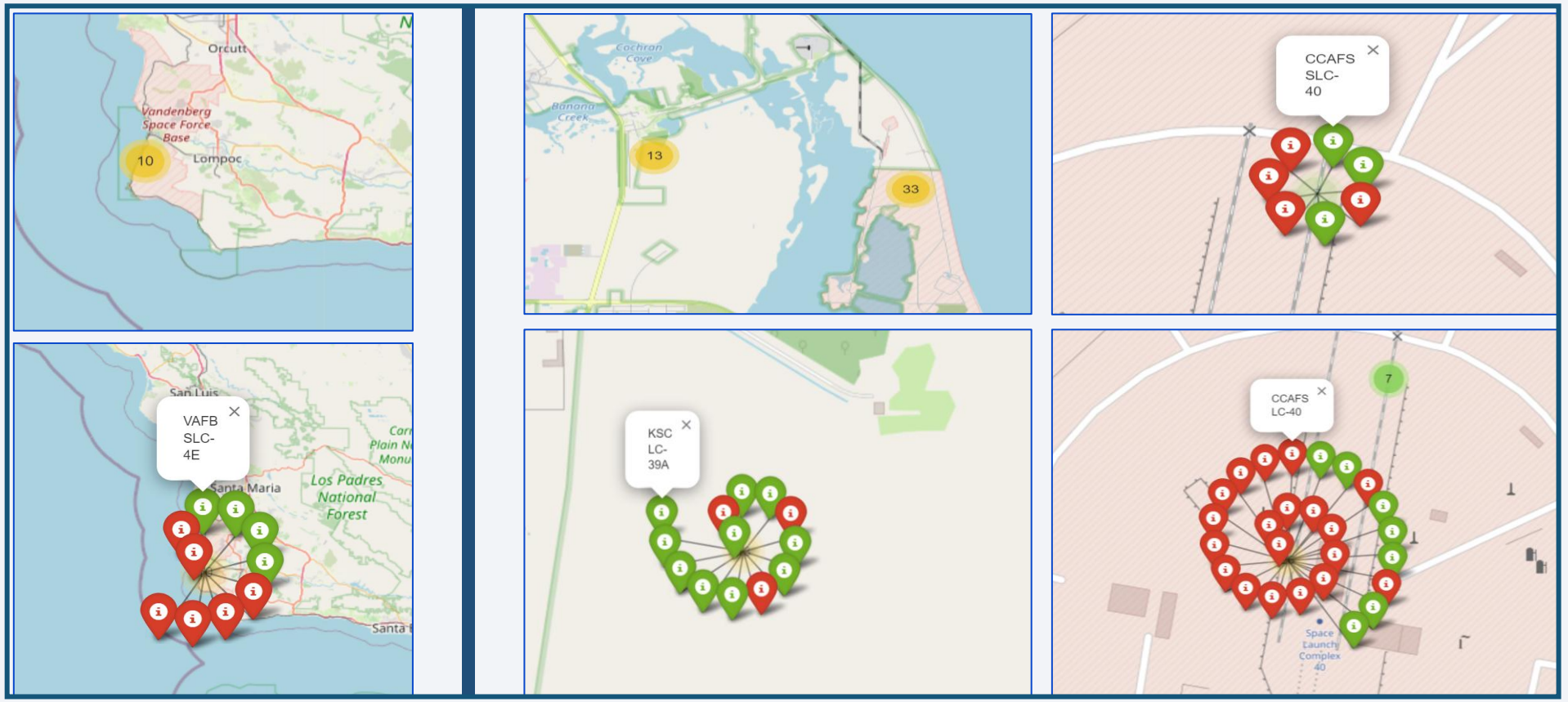
Coastal Launch Sites

Launch Sites are Located on East and West Coasts of the US

Launch Sites with Color Coding

West Coast Launch Site

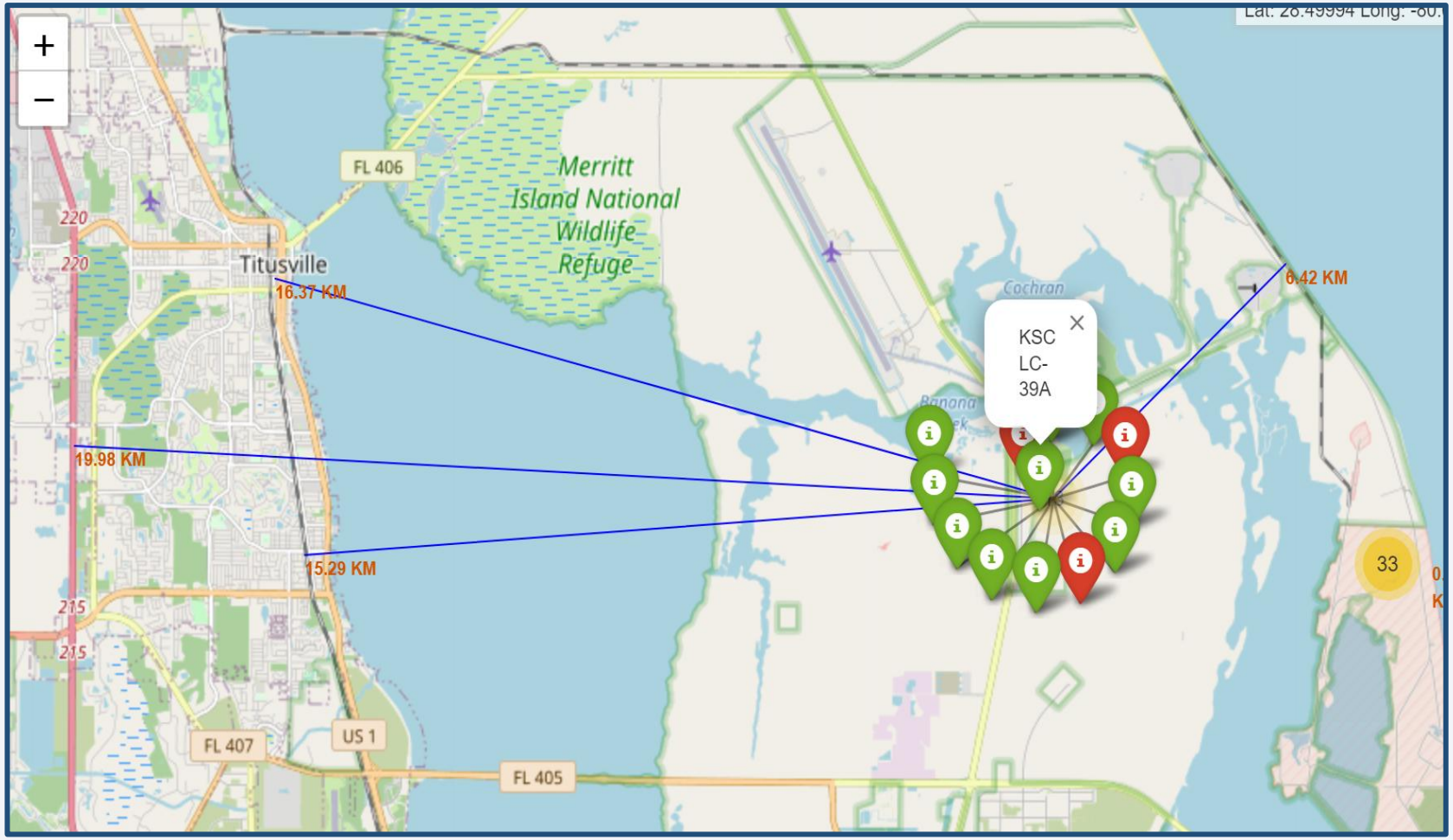
East Coast Launch Sites



Successful Launches have Green Markers

Unsuccessful Launches have Red Markers

Launch Site Proximity Map



KSC LC-39A

- Proximity to railway = 15KM
- Proximity to highway = 20KM
- Distance to nearest city =16KM
- Proximity to coastline = 8KM

Launch Sites

Are not in close proximity to:

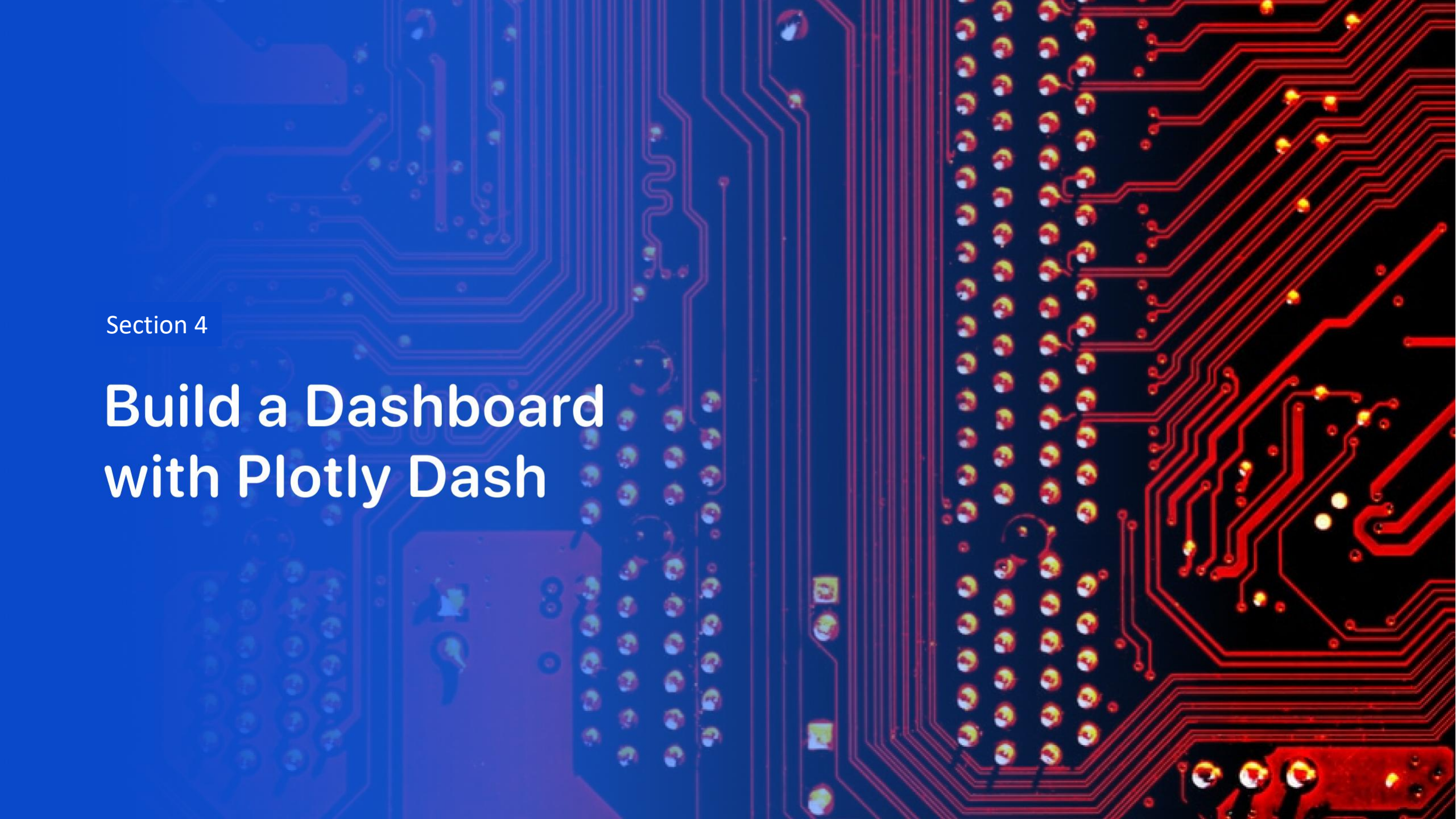
- Highways
- Railways
- Cities

Are in close proximity to:

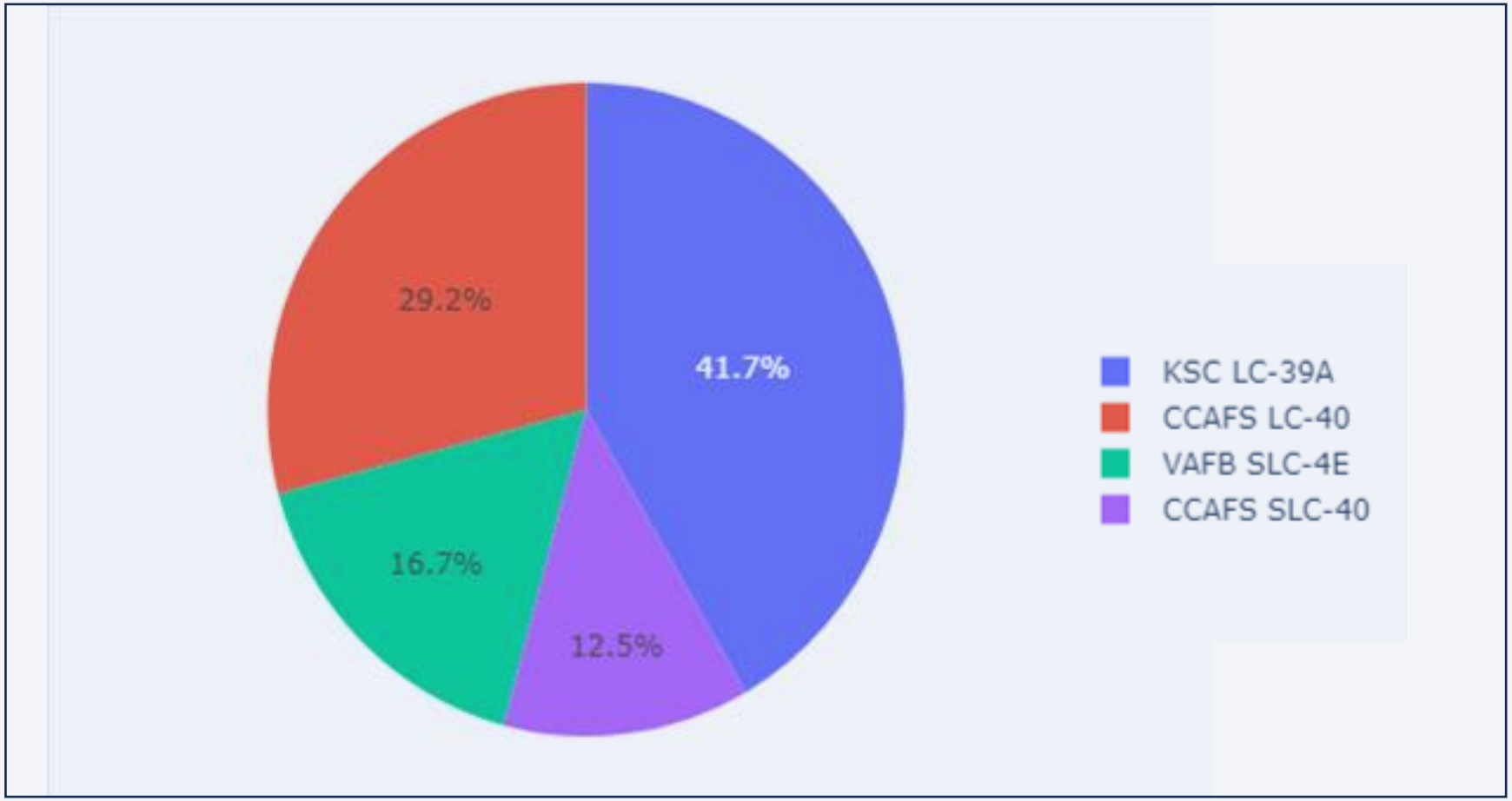
- Coastlines

Considerations

- Safety of Area Residents
- Safeguarding of infrastructure

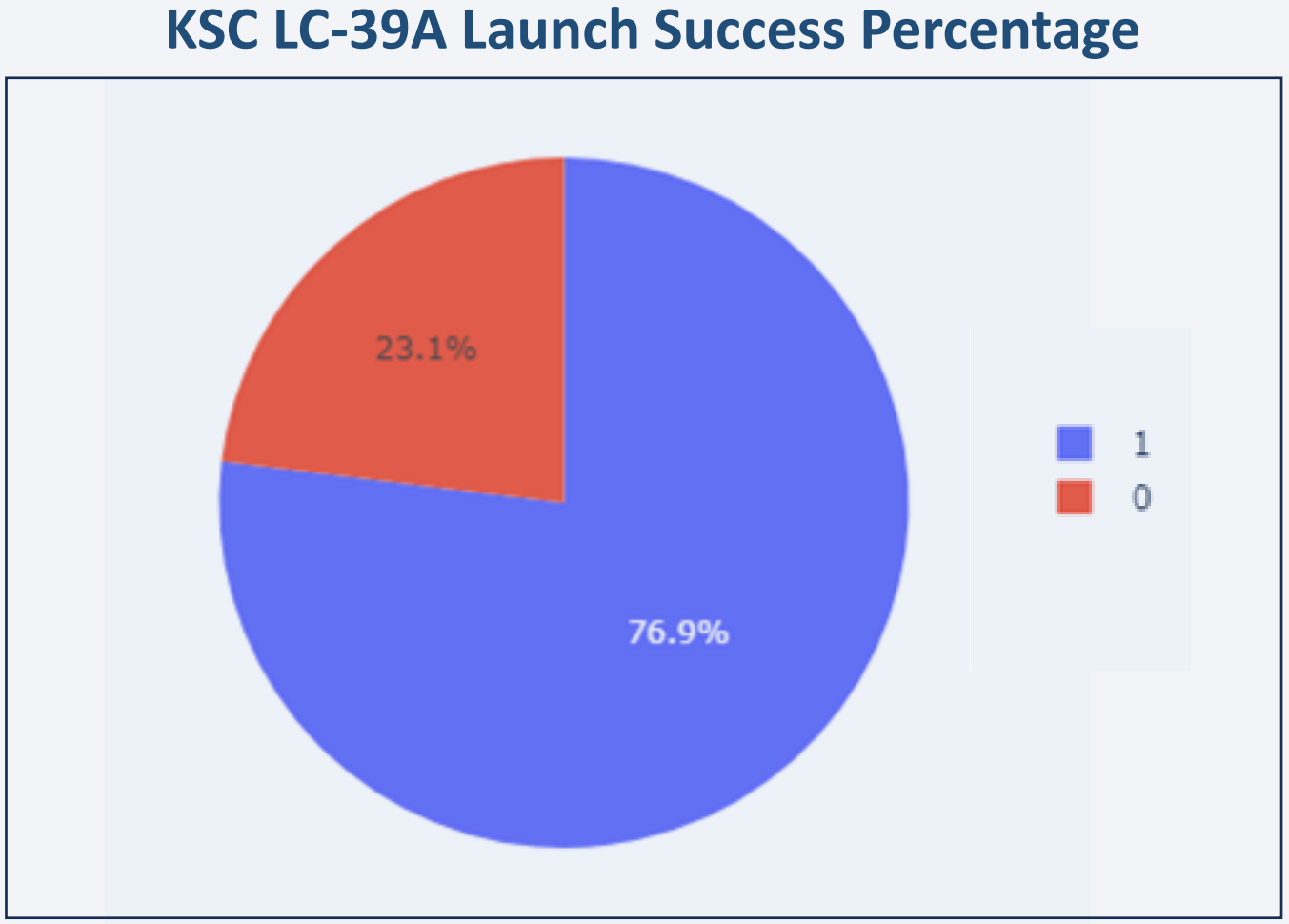

Section 4
Build a Dashboard with Plotly Dash

Success Counts by Launch Site



KSC LC-39A Has the Highest Count of Successful Launches

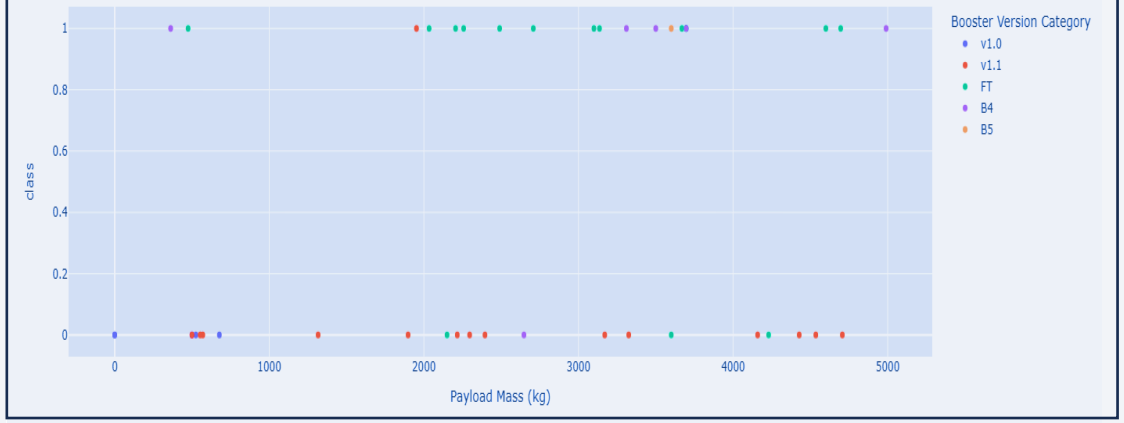
Launch Site with Highest Success Rate



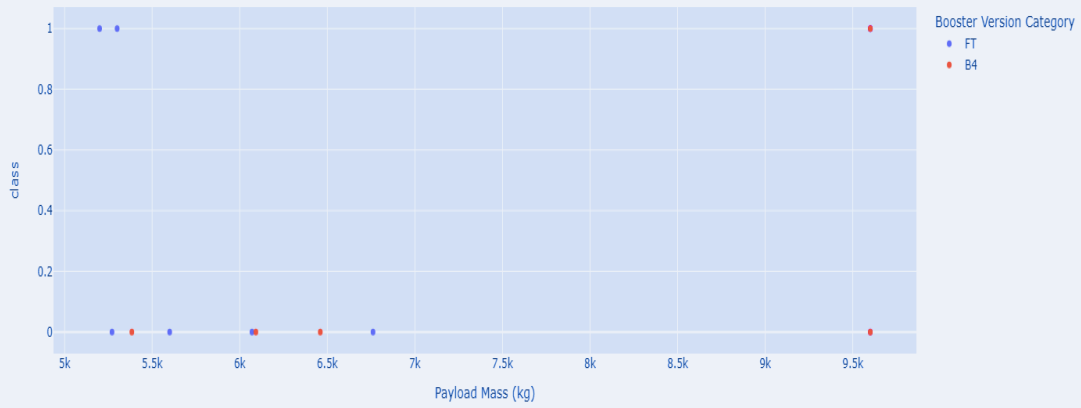
KSC LC-39A Has a 76.9% Launch Success Rate

Success by Payload Mass

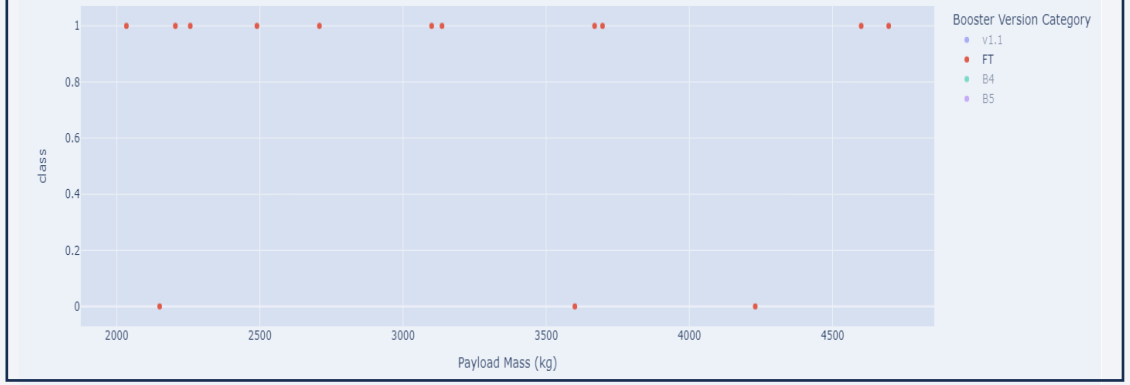
0 to 5,000K Payload



5,000 to 10,000K Payload



FT Version Success

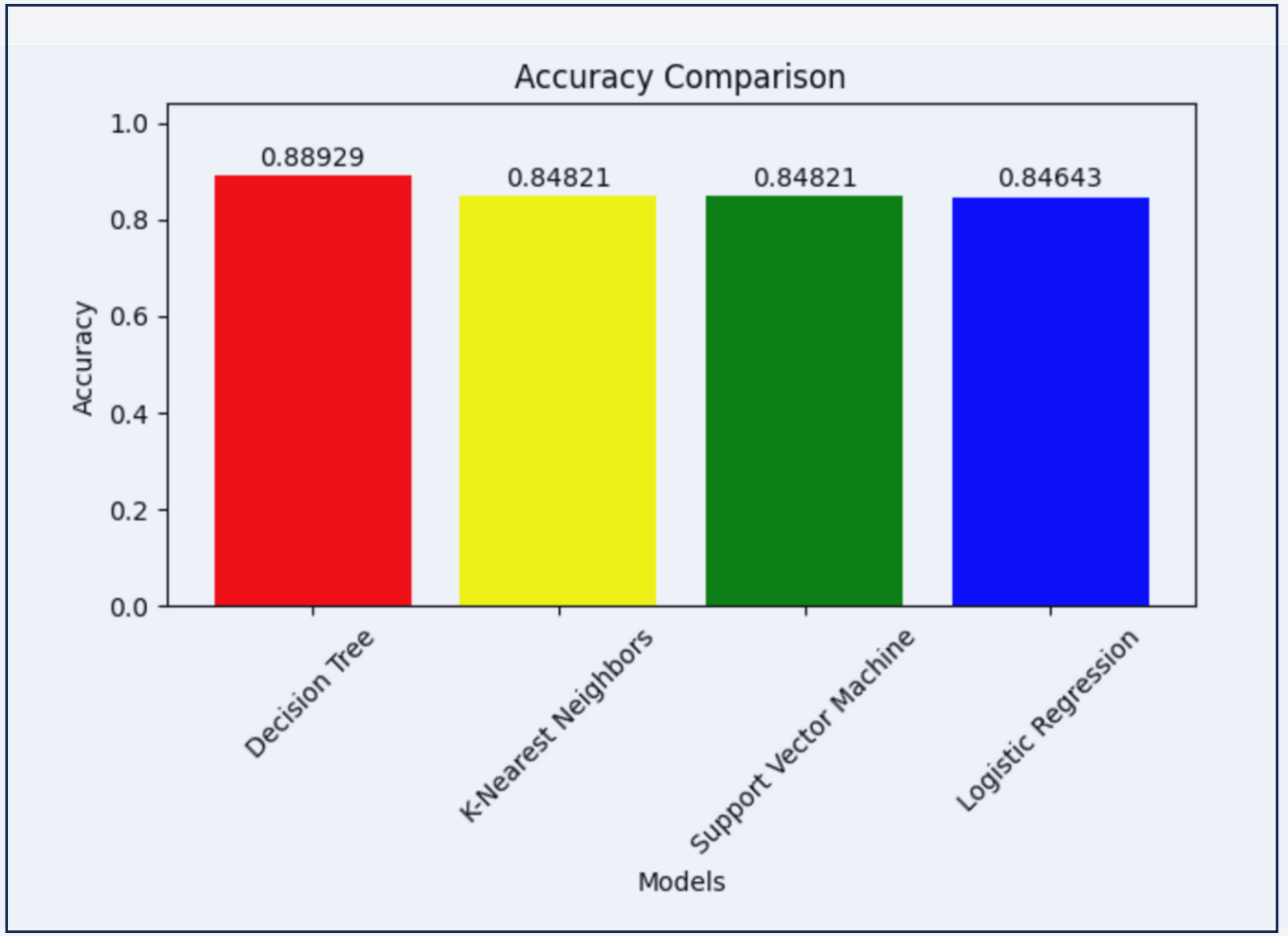


- Lighter Payloads have a Higher Success than Heavier Ones
- FT Version has a high success rate with payloads between 2,000 and 5,000K

Section 5

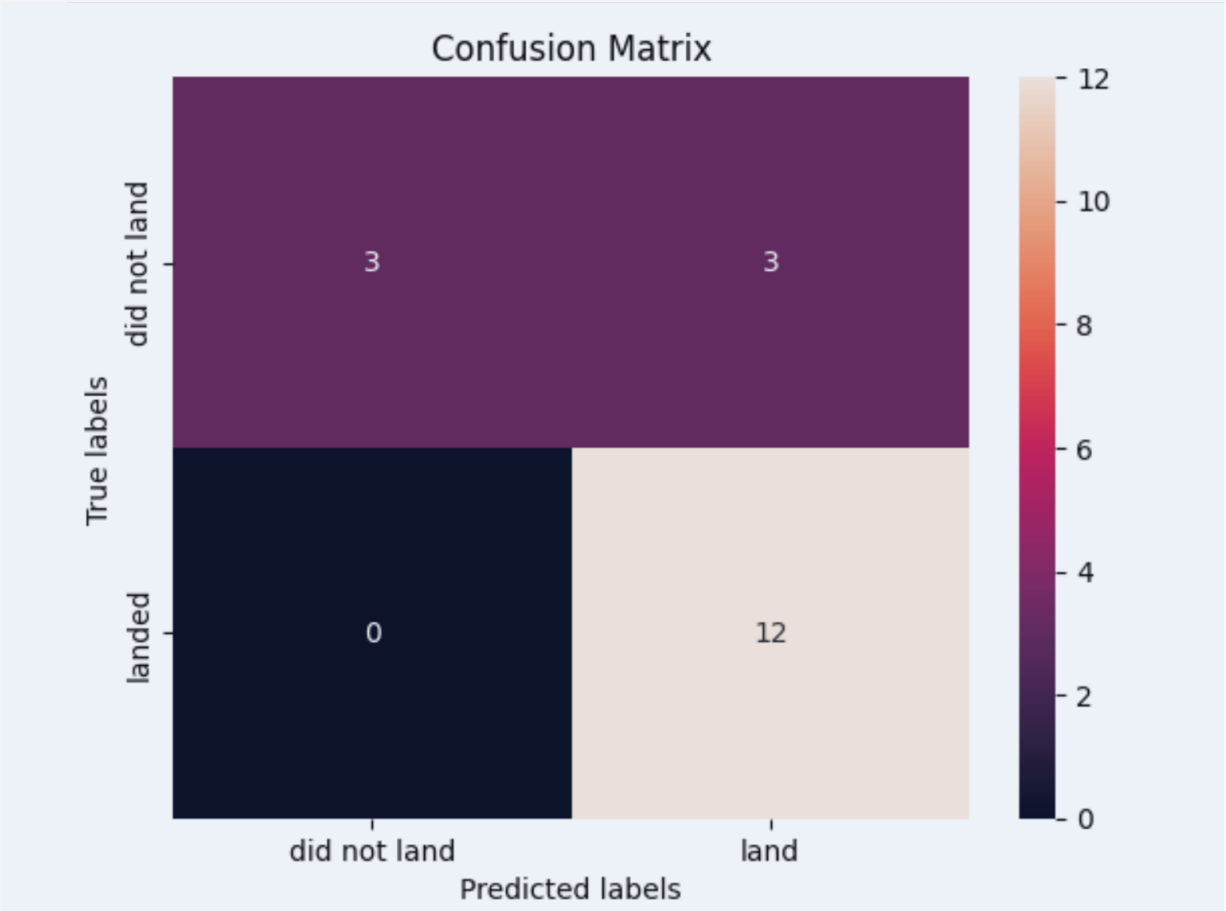
Predictive Analysis (Classification)

Classification Accuracy



Decision Tree has the Highest Accuracy and is the best Classification Model Choice

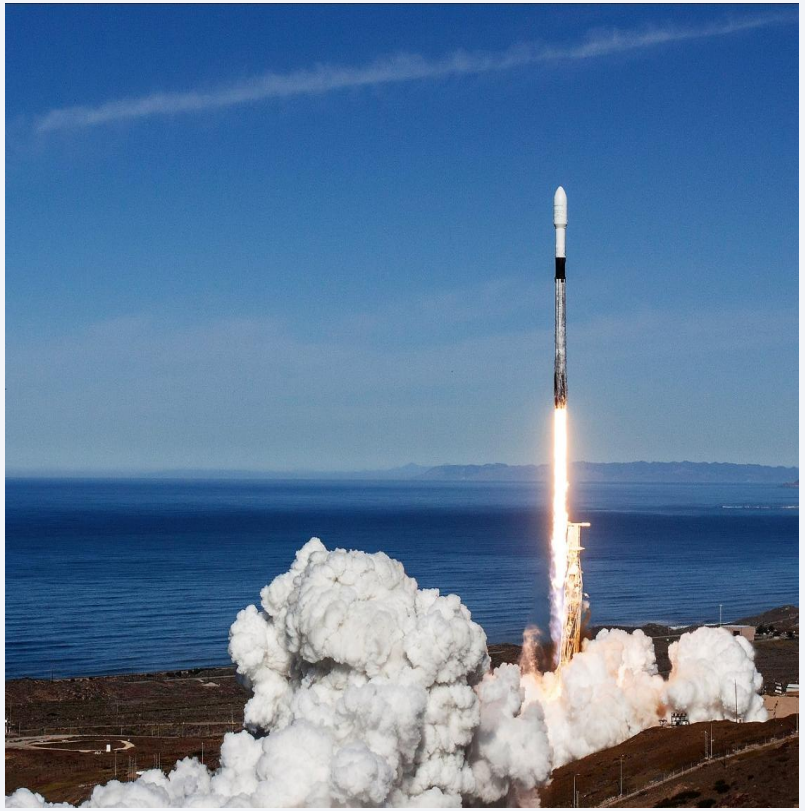
Confusion Matrix



The Decision Tree's Confusion Matrix Displays that the Model Struggles with False Positives

Conclusions

- **ES-L1, GEO, HEO & SSO Orbits have the highest success rates**
- **The KSC LC-39A Launch Site has the Highest Count of Successful Launches**
- **The Launch Success Rate rose Continually Between 2013 and 2020**
- **Launches with lighter Payloads have a higher probability of success than heavier ones**
- **FT Version has a high success rate with payloads between 2,000K and 5,000K**



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

Appendix - Page 1

Decision Tree Classifier Code

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import GridSearchCV

parameters = {
    'criterion': ['gini', 'entropy'],
    'splitter': ['best', 'random'],
    'max_depth': [2 * n for n in range(1, 10)],
    'max_features': ['sqrt'],  # Explicitly set max_features to 'sqrt'
    'min_samples_leaf': [1, 2, 4],
    'min_samples_split': [2, 5, 10]
}

tree = DecisionTreeClassifier(max_features='sqrt')  # Explicitly set max_features to 'sqrt'

tree_cv = GridSearchCV(tree, parameters, cv=10)
tree_cv.fit(X_train, Y_train)

print("Best parameters found: ", tree_cv.best_params_)
```

```
Best parameters found:  {'criterion': 'gini', 'max_depth': 10, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_spli
t': 5, 'splitter': 'random'}
```

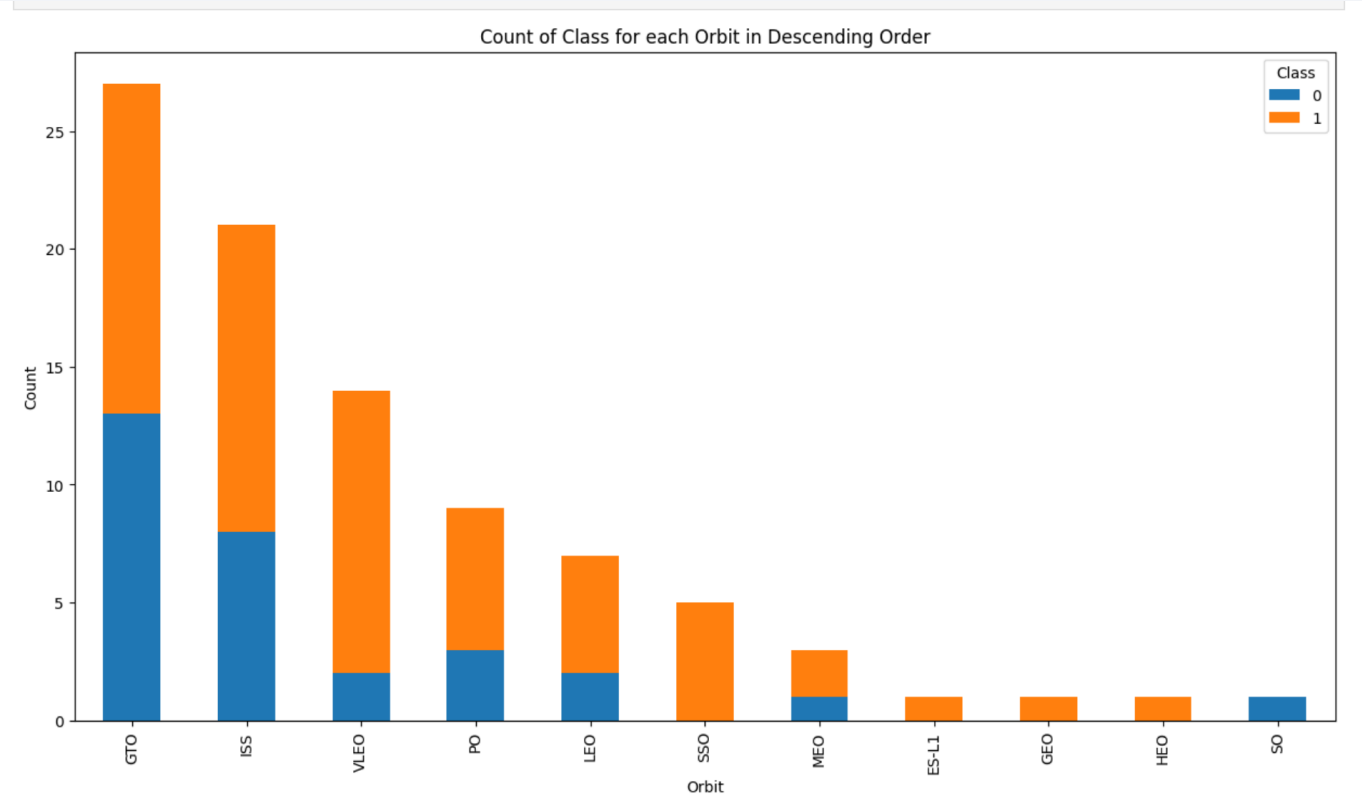
Appendix - Page 2

Folium Map Code

```
import folium

site_map = folium.Map(location=nasa_coordinate, zoom_start=5, tiles='Stamen Terrain')

launch_sites = {
    "CCAFS LC-40": [28.562302, -80.577356],
    "CCAFS SLC-40": [28.563197, -80.576820],
    "KSC LC-39A": [28.573255, -80.646895],
    "VAFB SLC-4E": [34.632834, -120.610745]
}

offsets = {
    "CCAFS LC-40": (0, 0),
    "CCAFS SLC-40": (0, 0),
    "KSC LC-39A": (0, 0),
}

for site, coordinates in launch_sites.items():
    folium.Circle(
        location=coordinates,
        radius=1000,
        color='#d35400',
        fill=True,
        fill_color='#e74c3c',
        fill_opacity=0.7,
        weight=1.5
    ).add_to(site_map)

    icon_anchor = offsets.get(site, (0, 0))
    folium.Marker(
        location=coordinates,
        icon=folium.DivIcon(
            html=f"""<div style="font-size: 5pt; color: #d35400;"><strong>{site}</strong></div>""",
            icon_anchor=icon_anchor
        )
    ).add_to(site_map)

site_map
```

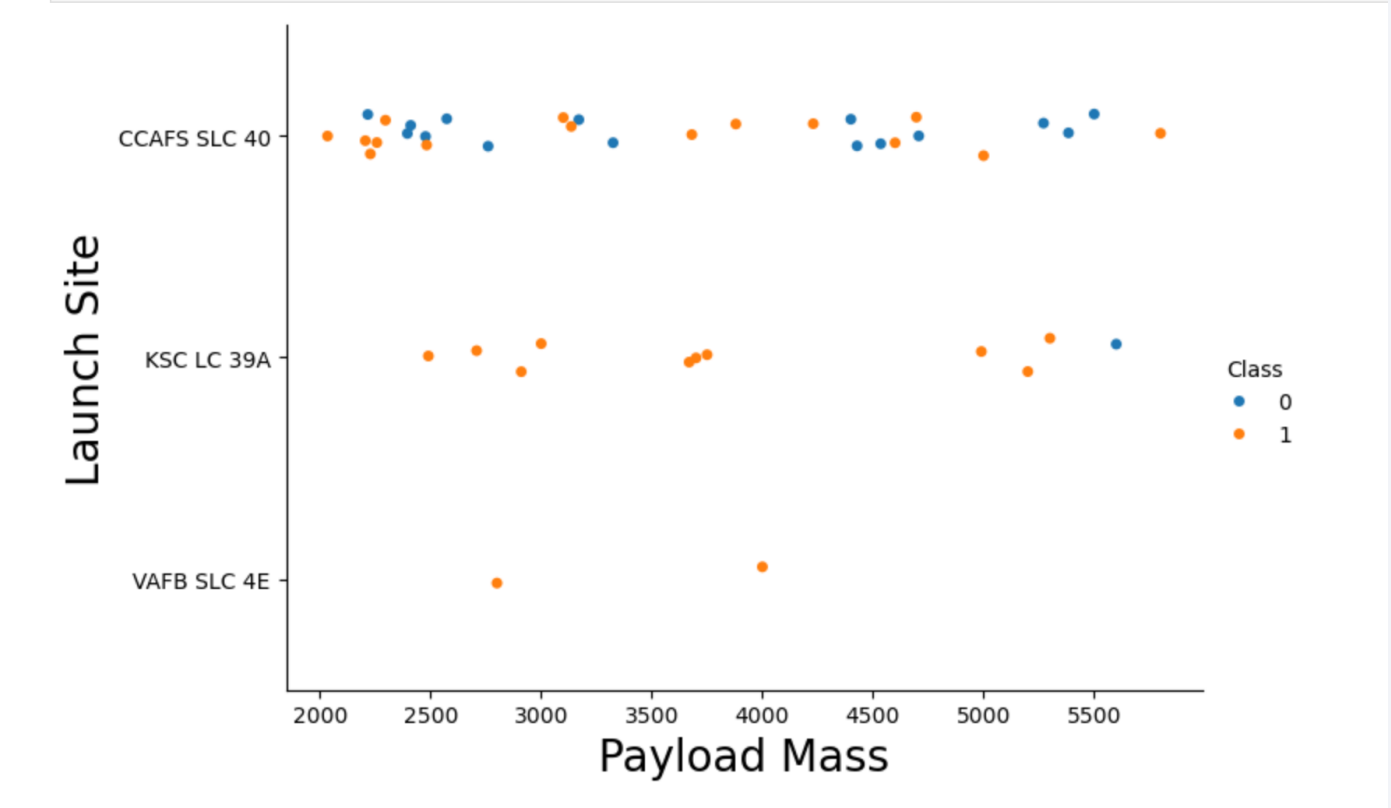
Appendix - Page 3

Orbit Success Stacked Bar



Appendix - Page 4

Launch Site Success - Payload Mass >= 2000K and <= 6000K



Thank you!