IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

A. S. M. Fazle Rabbi
25 July 2025

Outline

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

Executive Summary

This report explores the predictive factors contributing to the successful landing of the SpaceX Falcon 9 rocket's first stage. The analysis involved data collection through the SpaceX API and web scraping from Wikipedia, followed by data wrangling and exploratory analysis using Python and SQL. Various classification models Logistic Regression, KNN, Decision Tree, and SVM were implemented and evaluated. Among them, the Decision Tree classifier achieved the highest accuracy of 88.9%. The study also included interactive visualizations and dashboards to examine launch site performance, payload impact, and temporal trends. Insights from this analysis can support mission planning, improve risk assessment, and enhance operational decision-making for aerospace stakeholders.

Introduction

SpaceX has revolutionized the aerospace industry by drastically reducing the cost of launching payloads into space. The major innovation behind this cost reduction is the reuse of the Falcon 9 rocket's first stage. Traditionally, first stages are discarded after launch, but SpaceX retrieves and reuses them, saving millions per mission. A typical Falcon 9 launch costs around $62 million, in contrast to competitors whose costs can exceed $165 million.

The project involves collecting and processing SpaceX's launch data via an API and through web scraping of historical records. The data includes features such as booster version, payload mass, orbit type, launch site, and various landing parameters. Exploratory Data Analysis (EDA) and feature engineering are conducted to prepare the data for model training.

The goal of this project is to predict whether the Falcon 9 first stage will successfully land after launch. Such predictions are not only vital for cost estimation but also crucial for operational planning, risk mitigation, and competitive benchmarking. By building a machine learning model to predict landing success based on data from past launches, this project provides valuable insights for stakeholders, including commercial competitors and aerospace analysts.

Section 1

Methodology

Methodology

- Data collection methodology:
  - Data was collected using SpaceX API
  - Data was collected by web scraping from Wikipedia
- Perform data wrangling
  - Data was formatted into pandas data frame
  - Missing values was replaced using the mean values
  - Data was standardized for better performance

Methodology

- Perform exploratory data analysis (EDA) using visualization and SQL
  - Verities of plots were used to see the success rate, payload, and relation of features
  - SQL was used to query different insight from the database
- Perform interactive visual analytics using Folium and Plotly Dash
  - Launch sites and their success rate was visualized using Folium
  - Interactive dashboard was created to display success rate of the sites, and slider to analyze success rates based on the payload
- Perform predictive analysis using classification models
  - Logistic Regression, KNN, Decision Tree and SVM classification models were used.
  - GridSearchCV was used to tune the parameters
  - Accuracy Score and confusion matrix were used to evaluate the performance

Data Collection

1. Accessing SpaceX API
    - Used `requests.get()` to call SpaceX REST API endpoint “https://api.spacexdata.com/v4/launches/past”
    - Extracted launch metadata including rocket ID, launchpad ID, payload ID and core information
    - Applied helper functions to retrieve booster version, launch site, payload mass, orbit and core landing outcomes
2. Web Scraping from Wikipedia
    - Parsed launch tables from Wikipedia
    - Used BeautifulSoup to extract data fields such as date, launch site, payload, orbit and landing outcome

Data Collection – SpaceX API

Step 1: Initiate API Request

- Use the requests library to connect to the SpaceX API.
- Endpoint: https://api.spacexdata.com/v4/launches

Step 2: Parse API Response

- Convert API response from JSON to a Python dictionary.
- Extract relevant fields such as launch date, launch site, payload mass, rocket type, outcome.

Step 3: Store Data Locally

- Save extracted data into a pandas Data Frame.
- Store the Data Frame locally for further processing.

Github Link

Data Collection - Scraping

Step 1: Initiate Web Scraping

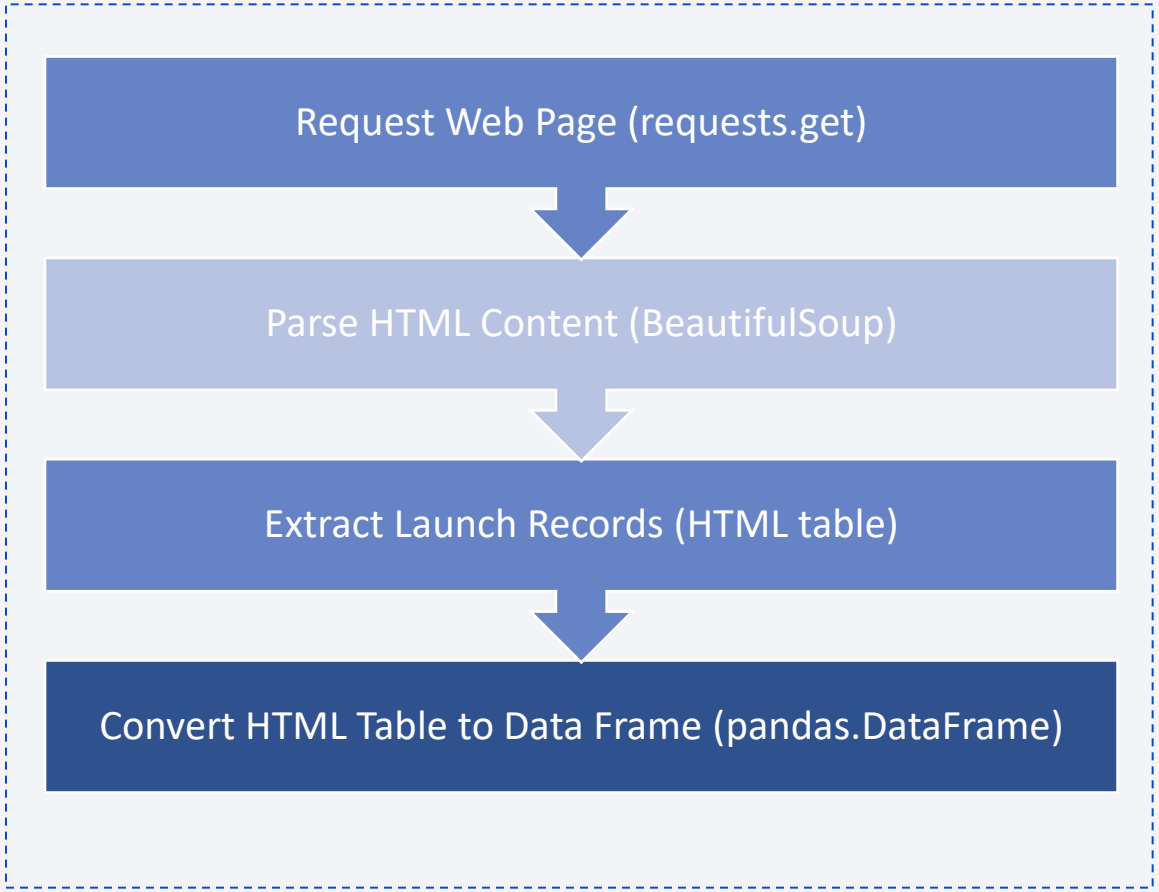
- Use the requests library to fetch the HTML content of the Wikipedia page.
- Target URL: https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches

Step 2: Parse HTML Content

- Use BeautifulSoup to parse the HTML content
- Extract the HTML table containing Falcon 9 launch records

Step 3: Convert to Data Frame

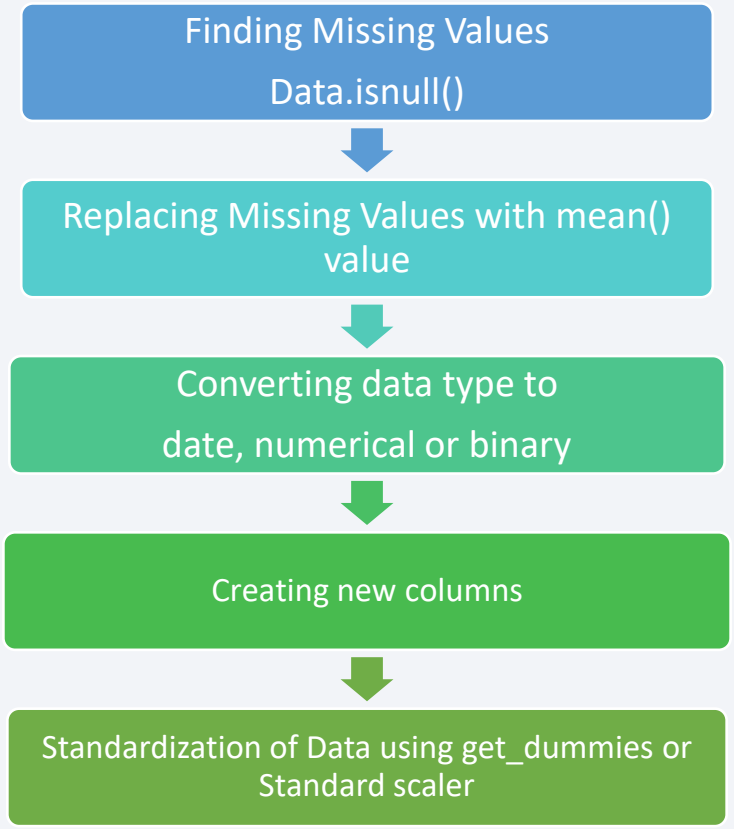
- Convert the extracted HTML table into a pandas Data Frame



Github Link

Data Wrangling

- Identify missing values in the data set.
- Replacing the missing values
- Convert data types to appropriate formats (e.g., date-time, numerical)
- Create new columns from the from the existing column (get dummies, marker color)
- Normalization/ Standardization features to ensure consistency



[Github Link]

EDA with Data Visualization

- Scatter plot:
  - Plotted Scatter plot to identify the relation between payload mass, flight number, launch site, and orbit.
- Line Graph:
  - Plotted line graph to see the trend of of success rate over the years
- Bar chart:
  - Plotted bar chart to see the success rate in different orbits as well as the performance of different classifier models
- Confusion Matrix :
  - Plotted confusion matrix to evaluate the performance of classifier models

Github Link

EDA with SQL

- Dropped the existing table if it already existed
- Created a new table from a CSV file and loaded it into the database
- Viewed the first few rows of the table to inspect the data
- Selected specific columns such as customer, booster version, and launch site
- Filtered records to show only successful launch outcomes
- Counted the number of successful launches
- Grouped the data by booster version and counted the number of launches for each version
- Sorted the data by payload mass in descending order
- Applied logical conditions to filter launches from a specific site that were successful
- Filtered records based on launch year using pattern matching on the date

Github Link

Build an Interactive Map with Folium

Map Objects Created and Added

- Markers
  - Used to label and identify different SpaceX launch sites on the map
  - Each marker included a popup or tooltip to show the name of the launch site
- Circles / Circle Markers
  - Used to visually emphasize each launch site
  - Helped indicate the area around the launch site, potentially useful for visual scale or distance
- Lines (Polylines)
  - Added between the launch site and the nearest city or coastline
  - Used to illustrate proximity and direction from the launch site to nearby landmarks

Purpose of Adding These Objects

- To visually represent SpaceX launch sites on a geographic map
- To enhance interactivity and clarity, showing additional information through popups
- To analyze distances between launch sites and important features like cities or coastlines
- To support spatial understanding of launch site distribution and accessibility

Github Link

Build a Dashboard with Plotly Dash

Plots/Graphs Added

1. Pie Chart (success-pie-chart)
    - Purpose (All Sites): Displays the distribution of successful launches across all launch sites.
    - Purpose (Single Site): Shows the proportion of successful vs. failed launches at the selected site.
    - Dynamic Behavior: Changes based on the dropdown selection of the launch site.
2. Scatter Plot (success-payload-scatter-chart)
    - Purpose: Illustrates the correlation between payload mass and launch success, with color coding by booster version.
    - Dynamic Behavior: Updates based on both the selected launch site (via dropdown) and the selected payload range (via slider).

Interactive Elements Added

1. Launch Site Dropdown
    - Allows the user to filter the dashboard by:
    - All Sites
    - Individual sites (CCAFS LC-40, KSC LC-39A etc.)
    - Used to dynamically update both the pie chart and scatter plot.
2. Payload Range Slider
    - Allows the user to specify a payload mass range (0 to 10,000 kg).
    - Filters the scatter plot to show only data within the selected range.

Github Link

Build a Dashboard with Plotly Dash

Why These Were Added

- Pie Chart: Helps users quickly assess launch performance either across all sites or for a specific site.
- Scatter Plot: Allows exploration of how payload mass correlates with mission success, highlighting patterns across different boosters.
- Dropdown + Slider Interactivity: Empowers users to interactively explore how success rates vary by site and payload mass useful for analysis and insight discovery.

Github Link

Predictive Analysis (Classification)

Model Development Process:

1. Data Preparation

- Data Loading: Read CSV data from a remote source
- Data Cleaning: Checked for missing values and duplicates
- Feature Engineering: Encoded categorical features and created a target label (Class)

2. Preprocessing

- Standardization: Applied “StandardScaler” to normalize features.
- Train-Test Split: Split data (80% training, 20% testing).

3. Model Training & Tuning

- Used “GridSearchCV” with 10-fold cross-validation for hyperparameter tuning.

4. Evaluation

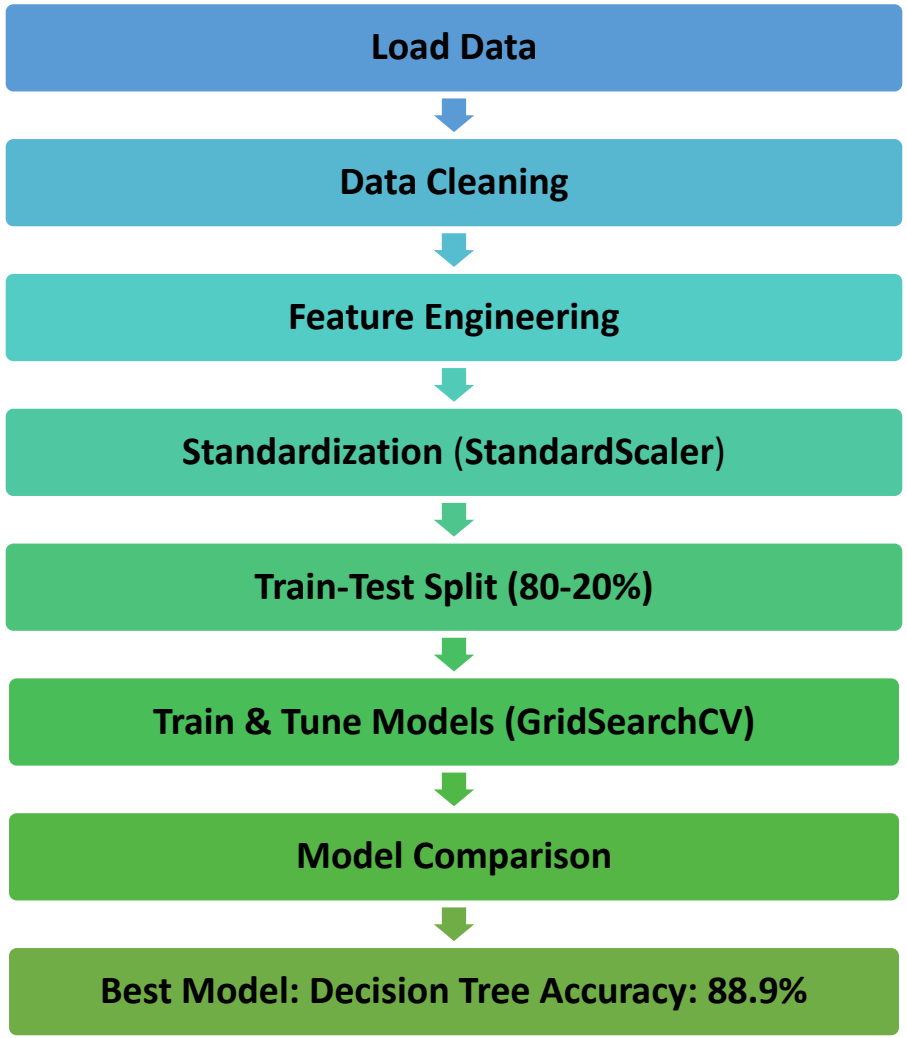
- Used accuracy score and confusion matrix to evaluate performance on test data.
- All models were assessed on:
    - Accuracy
    - Precision (via confusion matrix)

5. Model Selection

- Best Performing Model: Decision Tree
- Test Accuracy: 88.9%

Github Link

Predictive Analysis (Classification)



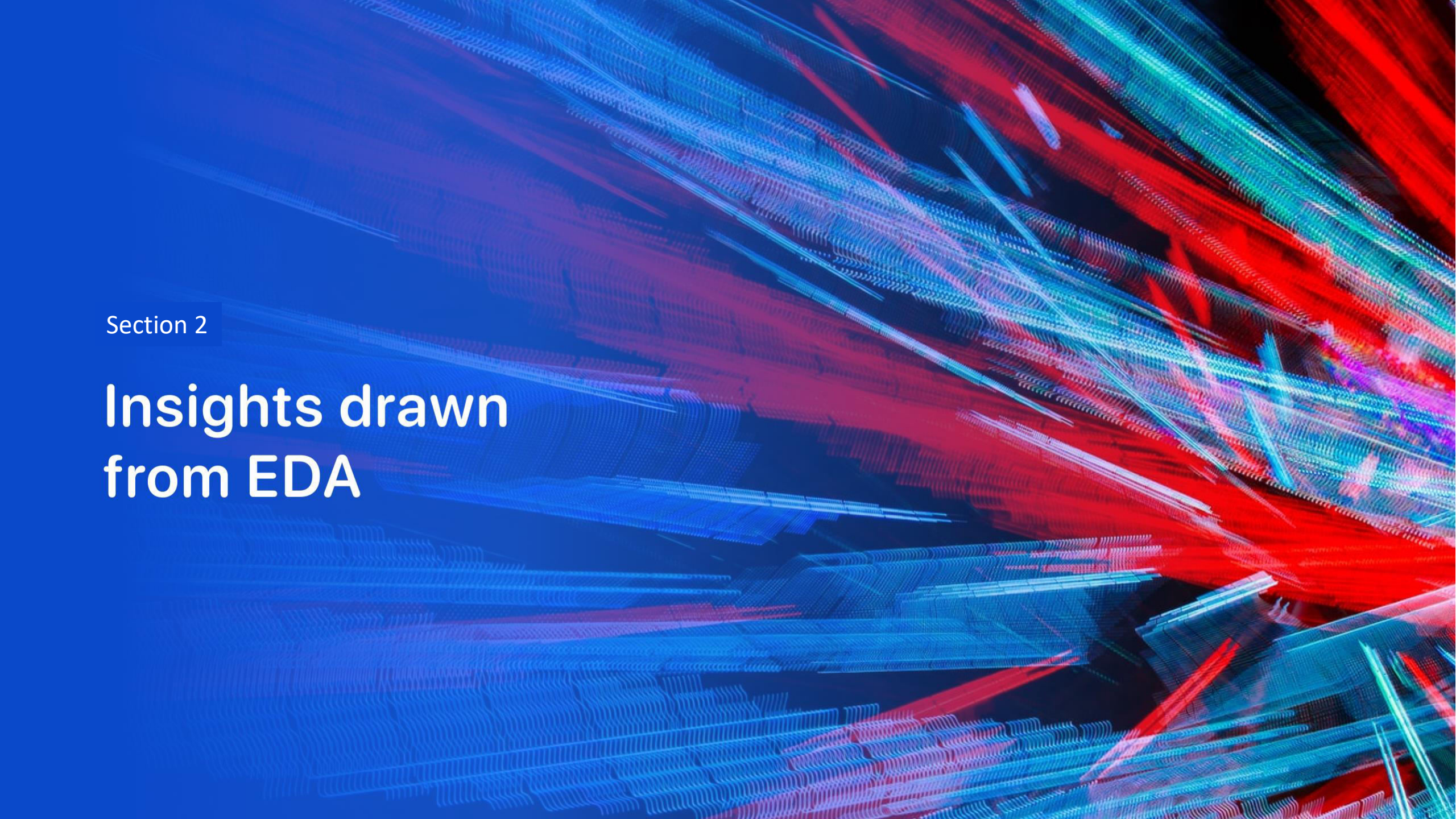
Github Link

Results

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

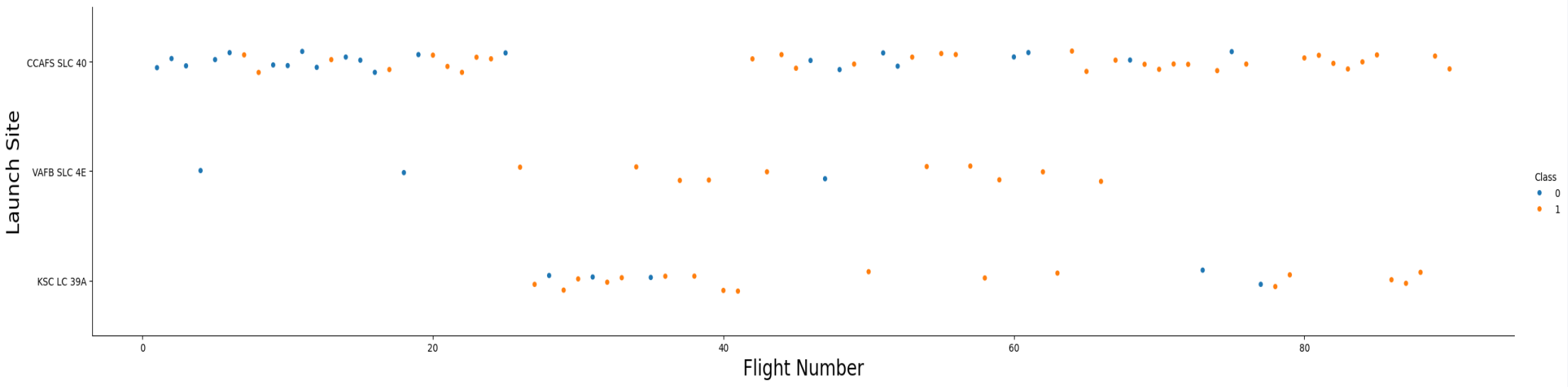
Section 2

Insights drawn from EDA



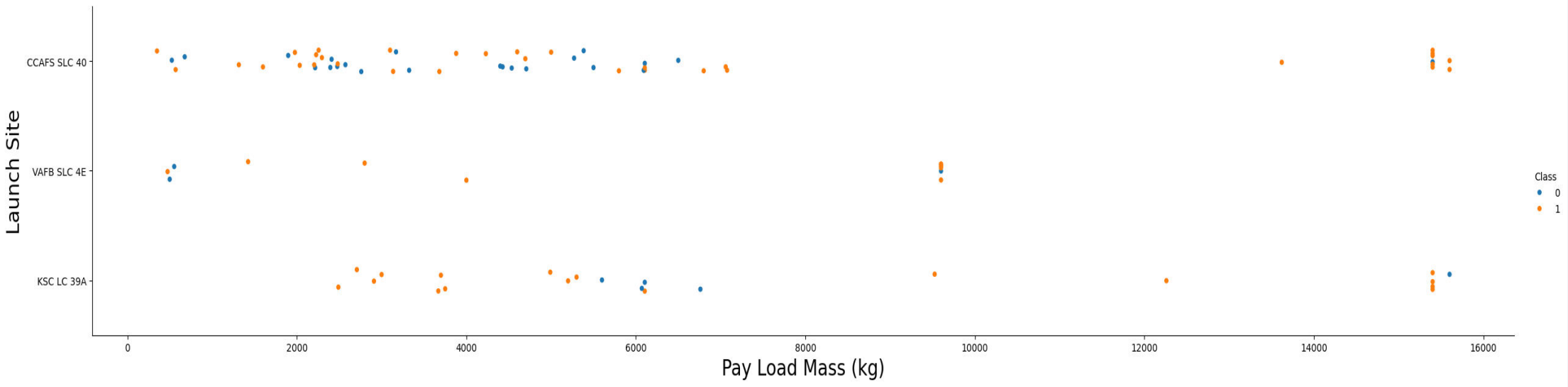
Flight Number vs. Launch Site

- Compares launch success over time across different sites.
- Some launch sites (like CCAFS SLC 40) have more frequent and successful launches.
- Earlier flights had more failed attempts, especially from newer or less-used sites.



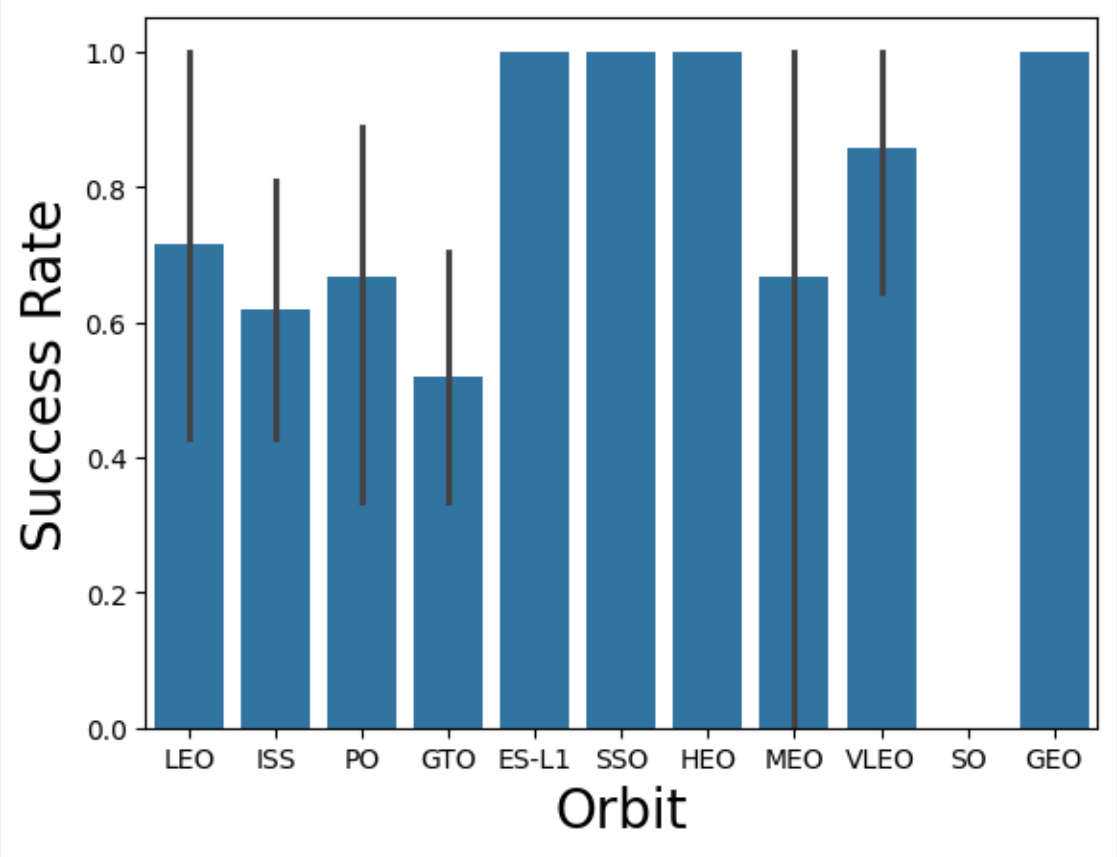
Payload vs. Launch Site

- Displays payload sizes launched from each site.
- VAFB-SLC 4E site doesn't handle very heavy payloads (>10,000 kg).
- CCAFS SLC 40 handles a wider range of payloads and shows a balanced success record.



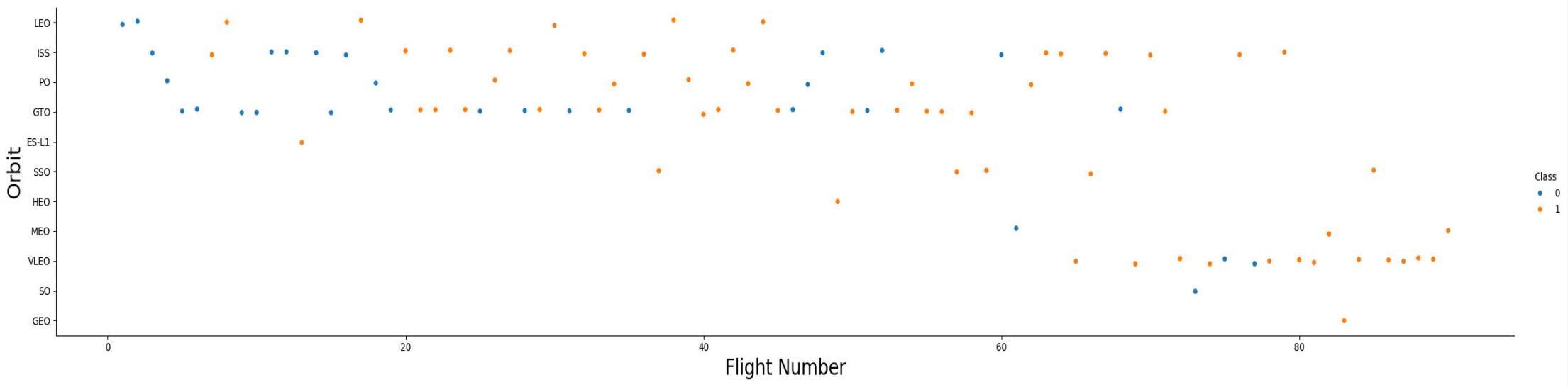
Success Rate vs. Orbit Type

- Aggregates success rates across different orbit types.
- ES-L 1, SSO, and HEO orbits have higher average success rates.
- SO orbit has no success rate, indicating greater risk.



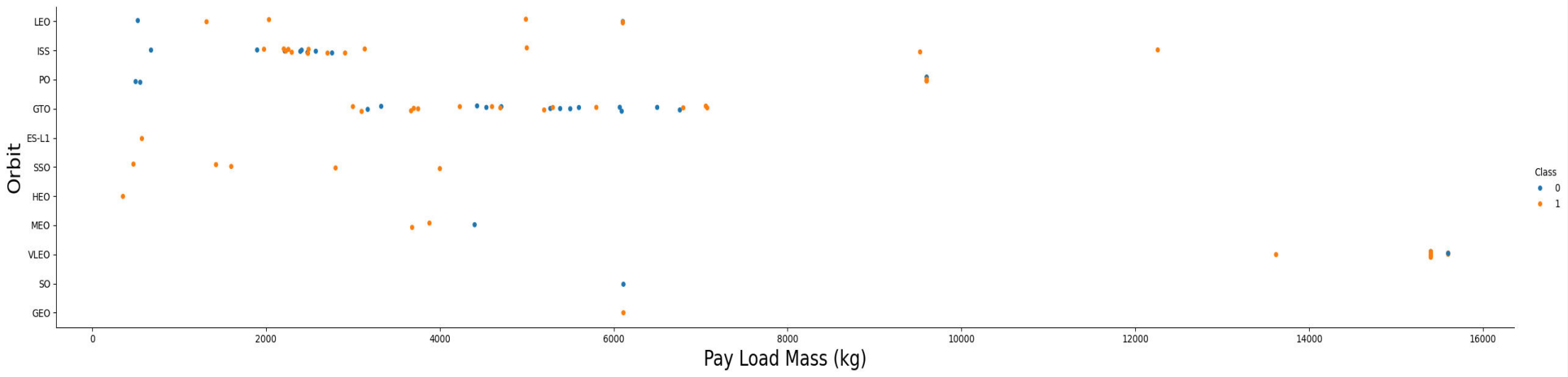
Flight Number vs. Orbit Type

- Explores how experience (flight number) affects success across orbit types.
- In LEO, success improves with flight number — a clear learning curve.
- GTO, ISS and Polar orbits show mixed or no clear trend with increasing flight numbers.



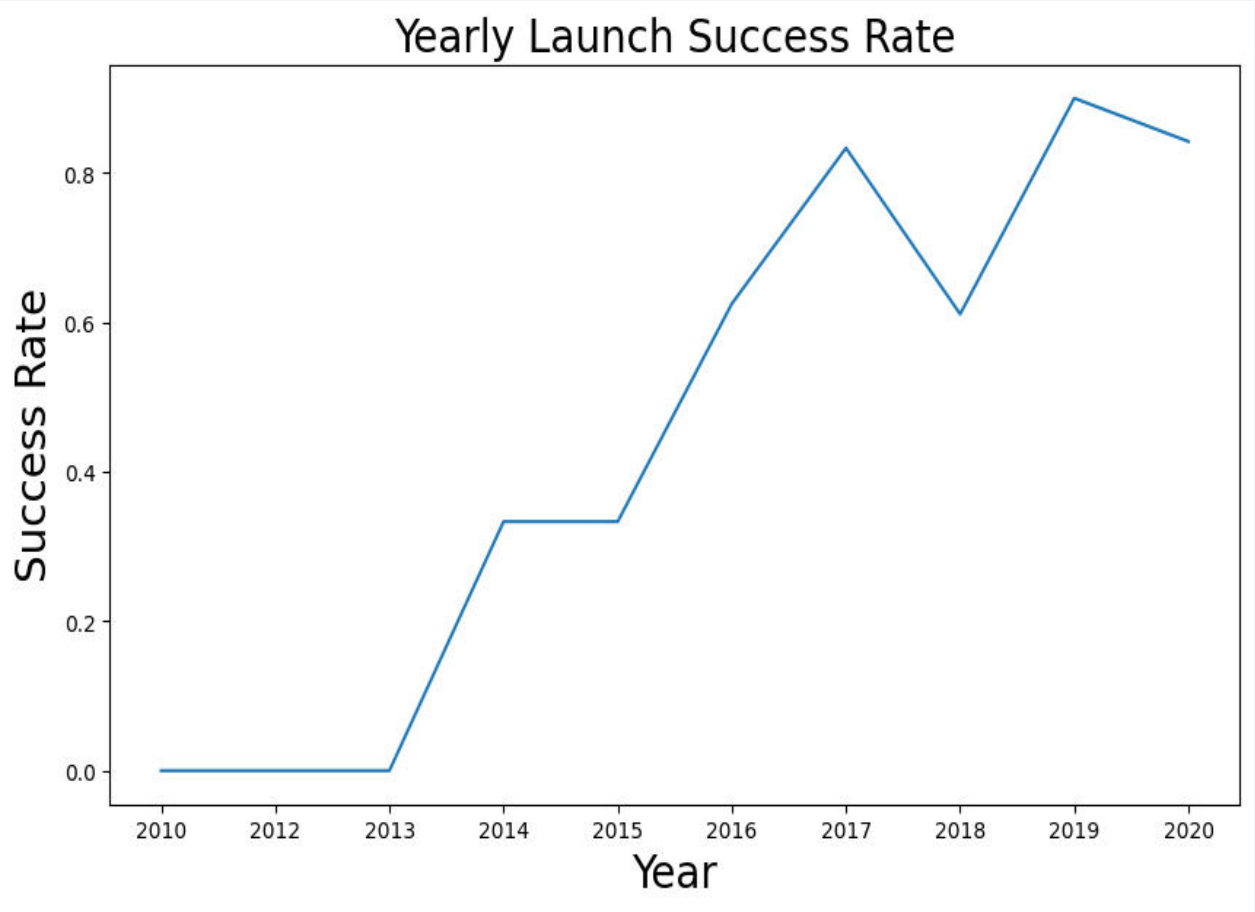
Payload vs. Orbit Type

- VLEO orbit handle heavier payloads more successfully.
- LEO and ISS orbits show successful landings even with heavy payloads.
- GTO shows success and failure across all payload weights, indicating less predictability.



Launch Success Yearly Trend

- Successful launch started in 2013
- After 2017, the success rate stabilizes near perfection, showing SpaceX maturity.
- Most successful year was 2019 with the success rate of almost 90%



All Launch Site Names

- Explanation: Retrieves all unique launch site names from the dataset, removing duplicates.

```
[ ]  %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
```

Launch Site Names Begin with 'CCA'

- Explanation: Fetches 5 records where the launch site starts with "CCA" (example: CCAFS LC-40), filtering based on the string pattern.

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

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

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- Explanation: Calculates the total payload mass (in kg) for launches where the customer was NASA (CRS).

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

```
 * sqlite:///my_data1.db
Done.
SUM(PAYLOAD_MASS__KG_)
45596
```

Average Payload Mass by F9 v1.1

- Explanation: Finds the average payload mass carried by the F9 v1.1 booster version.

```
%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
```

First Successful Ground Landing Date

- Explanation: Returns the earliest date of a successful landing on a ground pad.

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

```
 * sqlite:///my_data1.db
Done.
 MIN(Date)
 2015-12-22
```

Successful Drone Ship Landing with Payload between 4000 and 6000

- Explanation: Lists booster versions that landed successfully on a drone ship and carried payloads between 4000 and 6000 kg.

```
[ ] %sql SELECT Booster_Version FROM SPACEXTABLE WHERE Landing_Outcome = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000;
 * sqlite:///my_data1.db
Done.
```

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

Total Number of Successful and Failure Mission Outcomes

- Explanation: Counts how many missions had each type of mission outcome.

```
%sql SELECT Mission_Outcome, COUNT(*) FROM SPACEXTABLE GROUP BY Mission_Outcome;
```

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

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

Boosters Carried Maximum Payload

- Explanation: Finds booster versions that carried the maximum payload mass.

```
%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

2015 Launch Records

- Explanation: Lists all records in 2015 where the landing outcome was a failure on a drone ship, and extracts the month of launch.

```
[ ] %sql SELECT substr(Date, 6, 2) AS month, Date, Booster_Version, Launch_Site, Landing_Outcome FROM SPACEXTABLE WHERE  Landing_Outcome = 'Failure (drone ship
```

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

month	Date	Booster_Version	Launch_Site	Landing_Outcome
01	2015-01-10	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
04	2015-04-14	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

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

- Explanation: Ranks and counts different types of landing outcomes between June 4, 2010, and March 20, 2017, in descending order.

```
%sql SELECT Landing_Outcome, COUNT(*) AS count FROM SPACEXTABLE WHERE Date BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY Landing_Outcome ORDER BY count DESC;
```

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

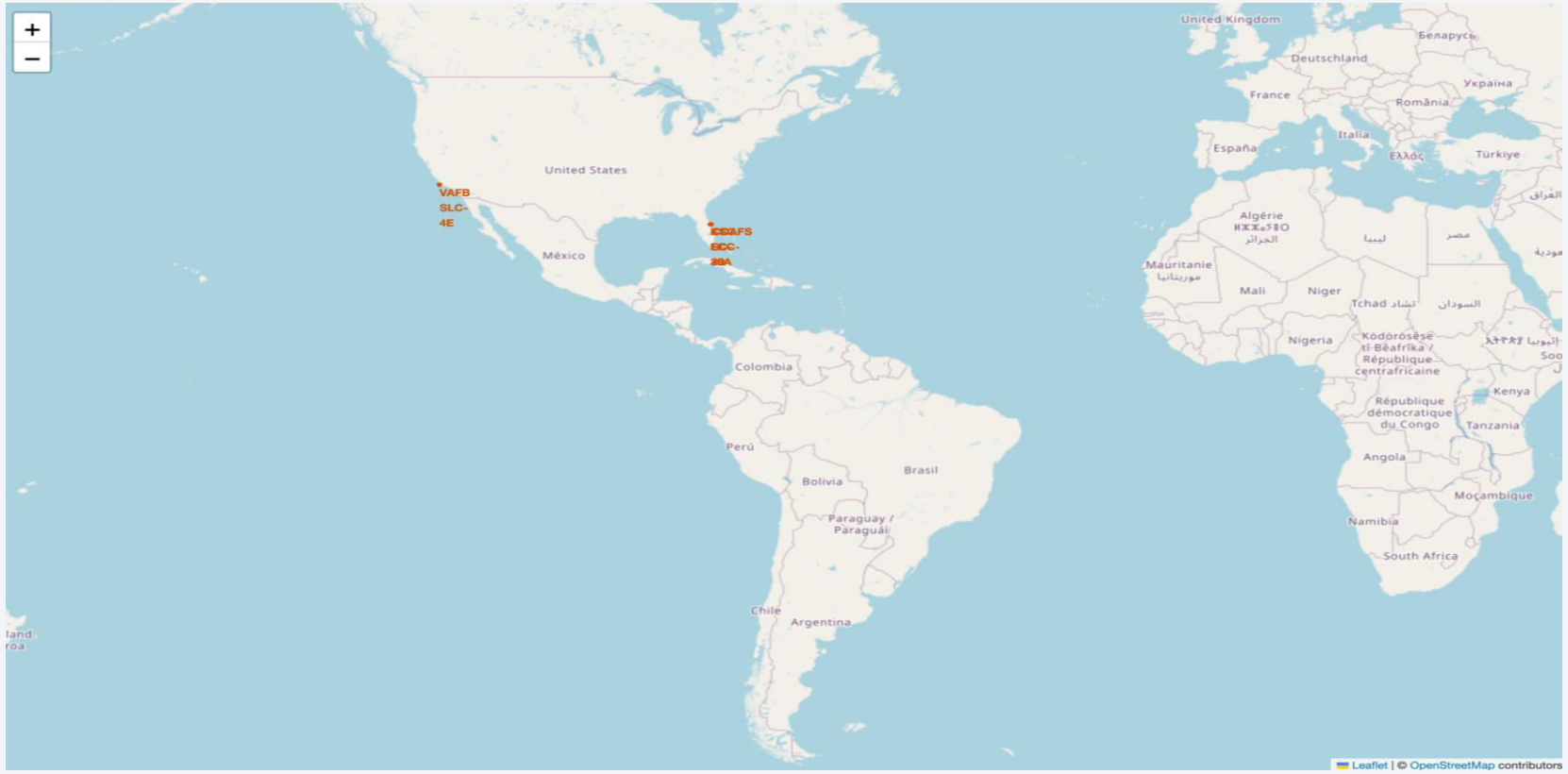
Landing_Outcome	count
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Section 3

Launch Sites Proximities Analysis

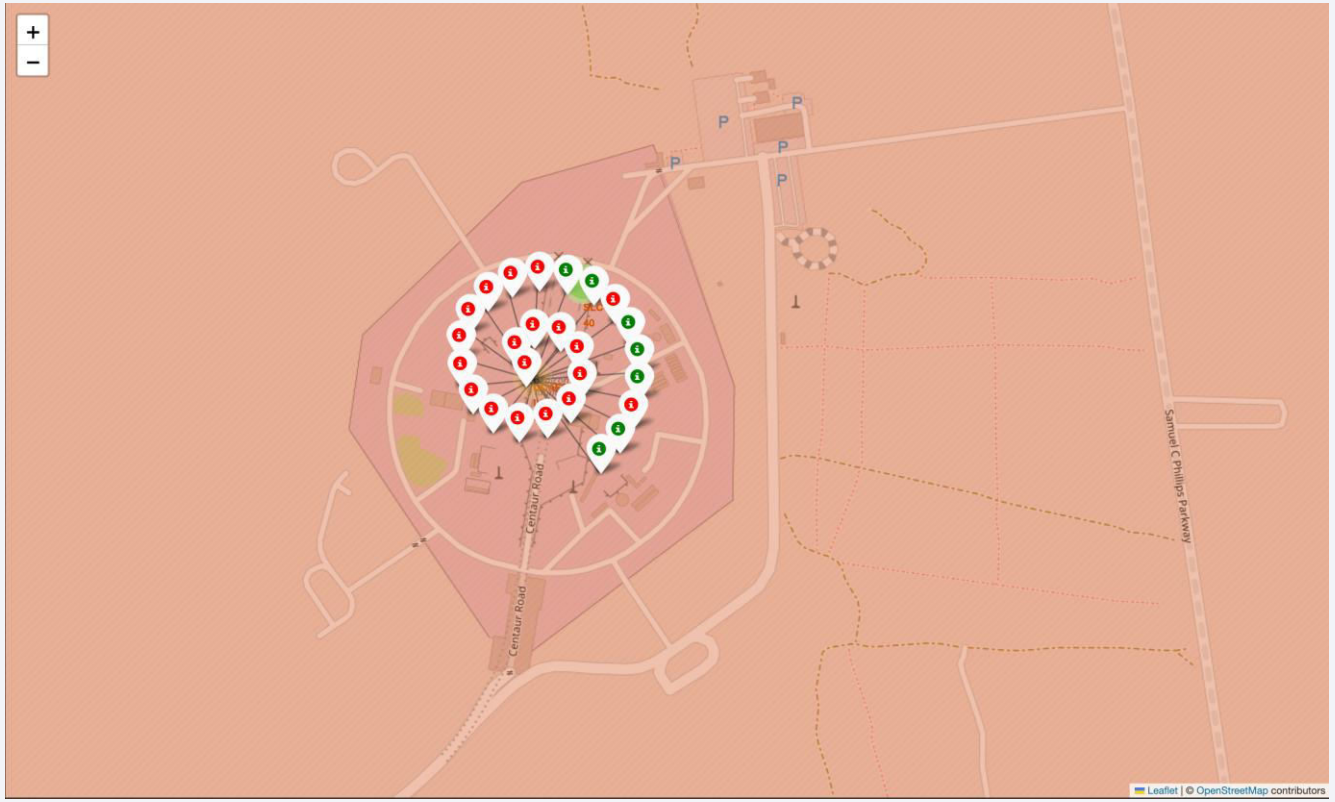
Launch Sites On World Map

- All launch sites are below the equator line
- All launch sites are close to the coast



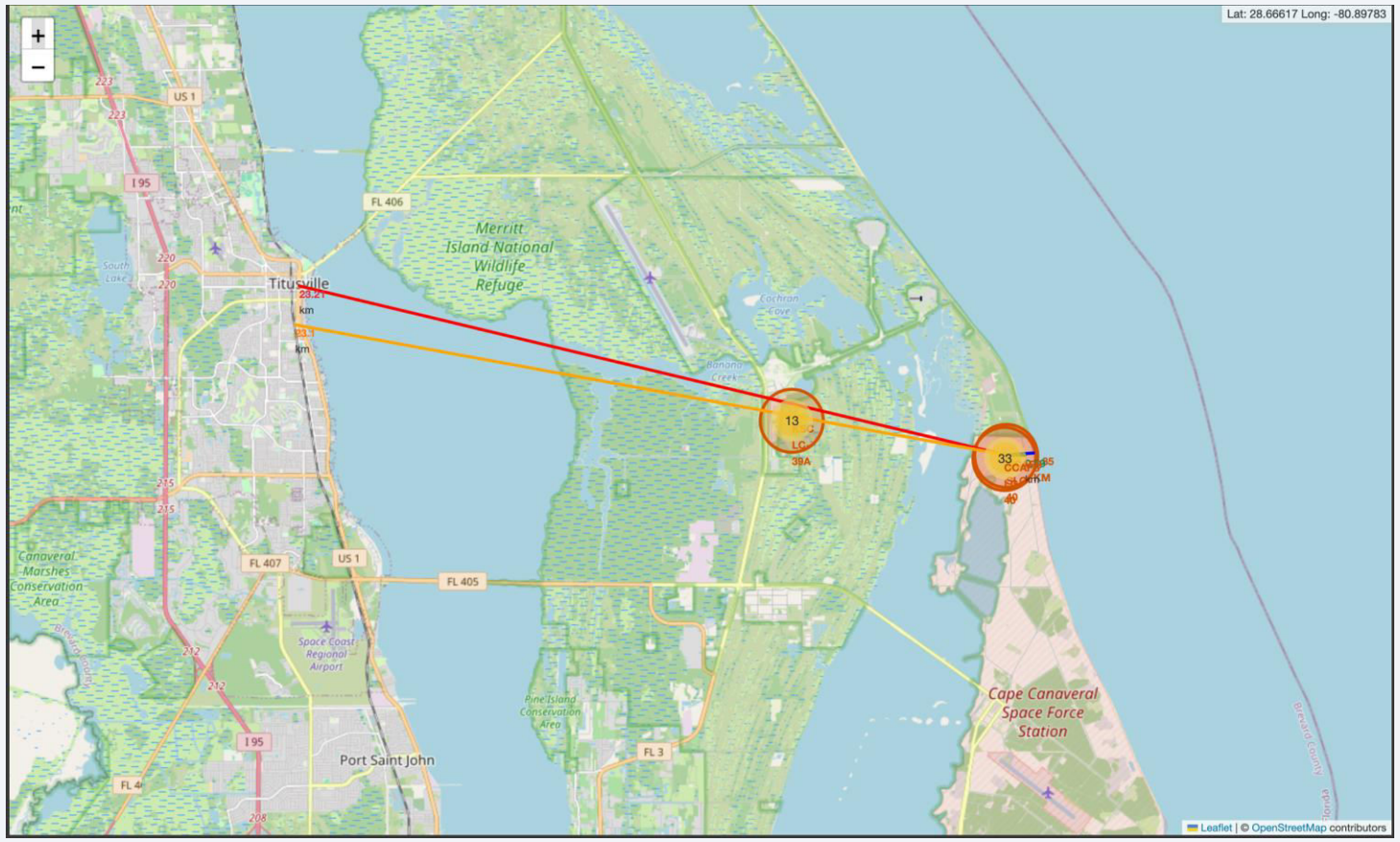
Launch Outcome of CCAFS LC -40

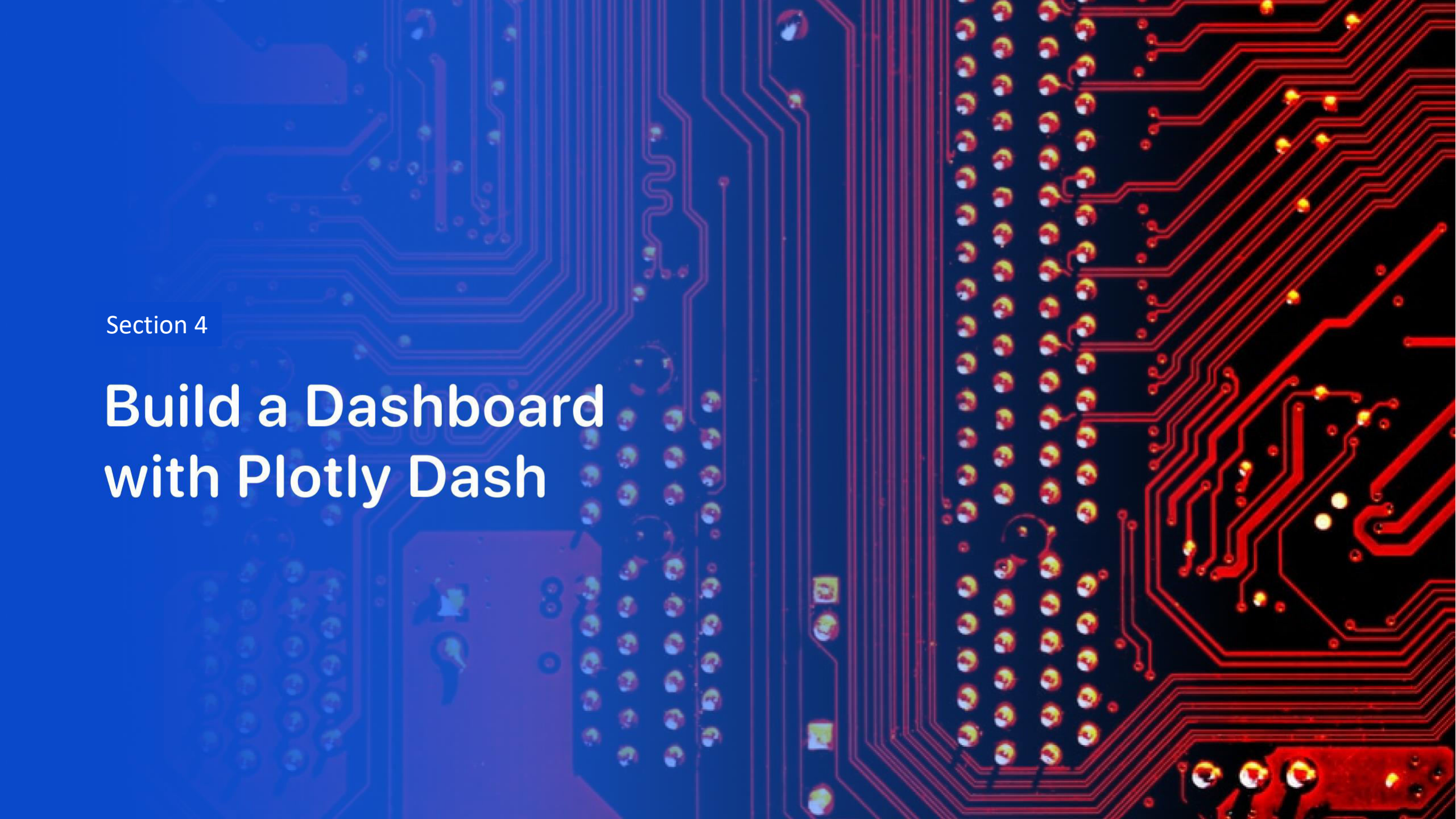
- Out of 26 launch 7 successful (green) 19 unsuccessful (red)



Distance from CCAFS SLC-40

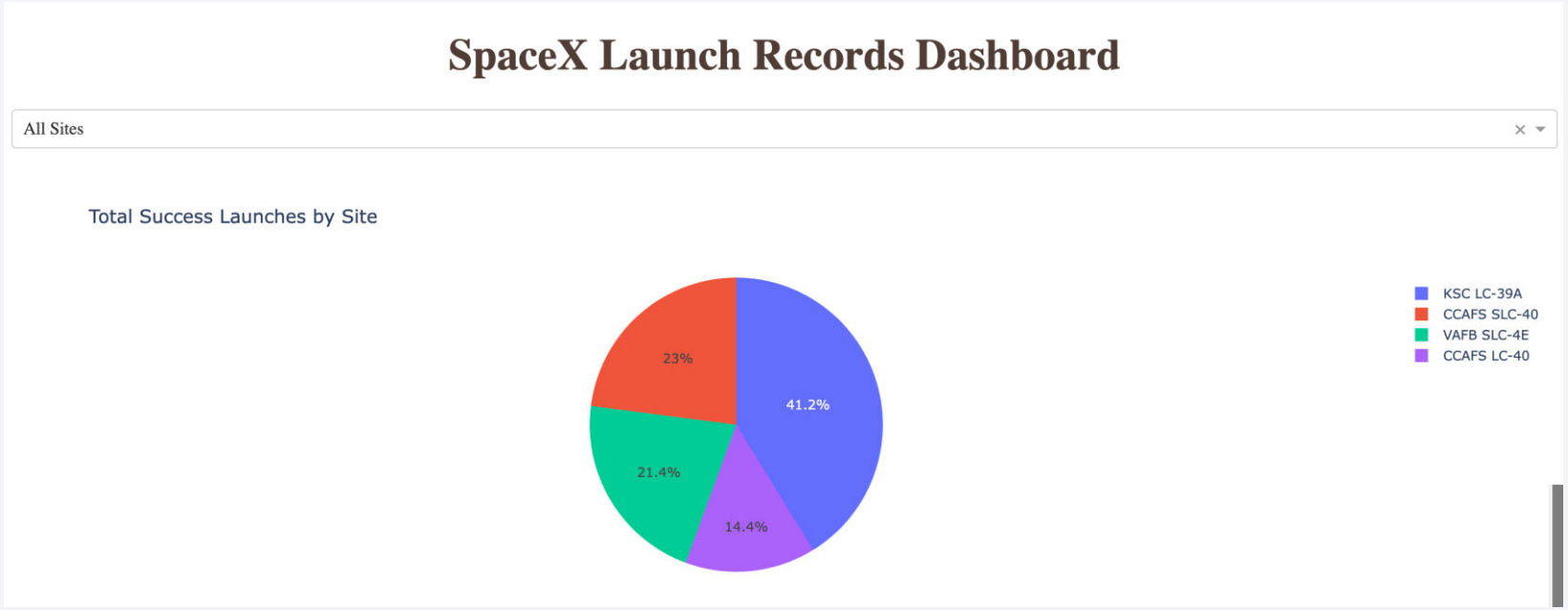
- Railway 23.1km (Yellow), Highway 0.59 km (Green), Coastline 0.85 km (Blue), City 23.21 km (Red)




Section 4
Build a Dashboard with Plotly Dash

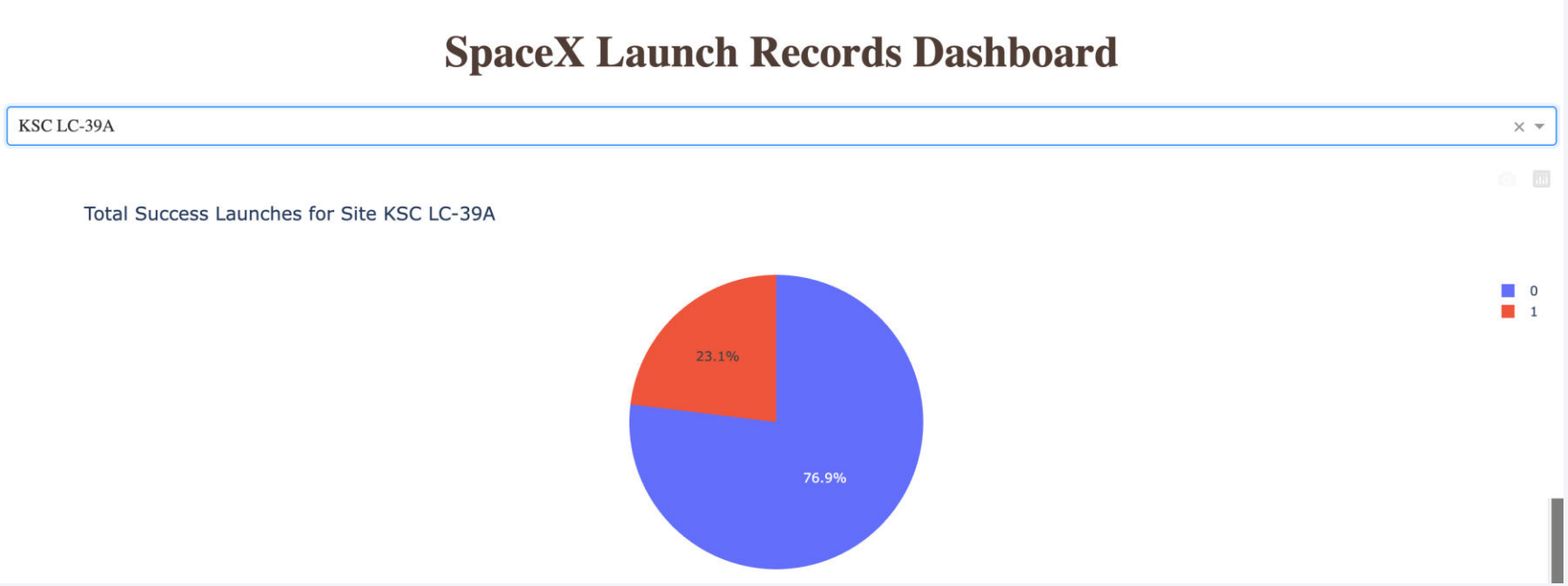
Success rate count of different launch sites

- KSC LC -39A has 41.2% (highest)
- CCAFS SLC -40 23% and VABF SLC -4E 21.4%
- CCAFS LC -40 14.4% (lowest)



Site with most successful launches

- KSC LC -39A is the site with most successful launches.
- 76.9% success rate



Correlation between payload (3000-8000) and success

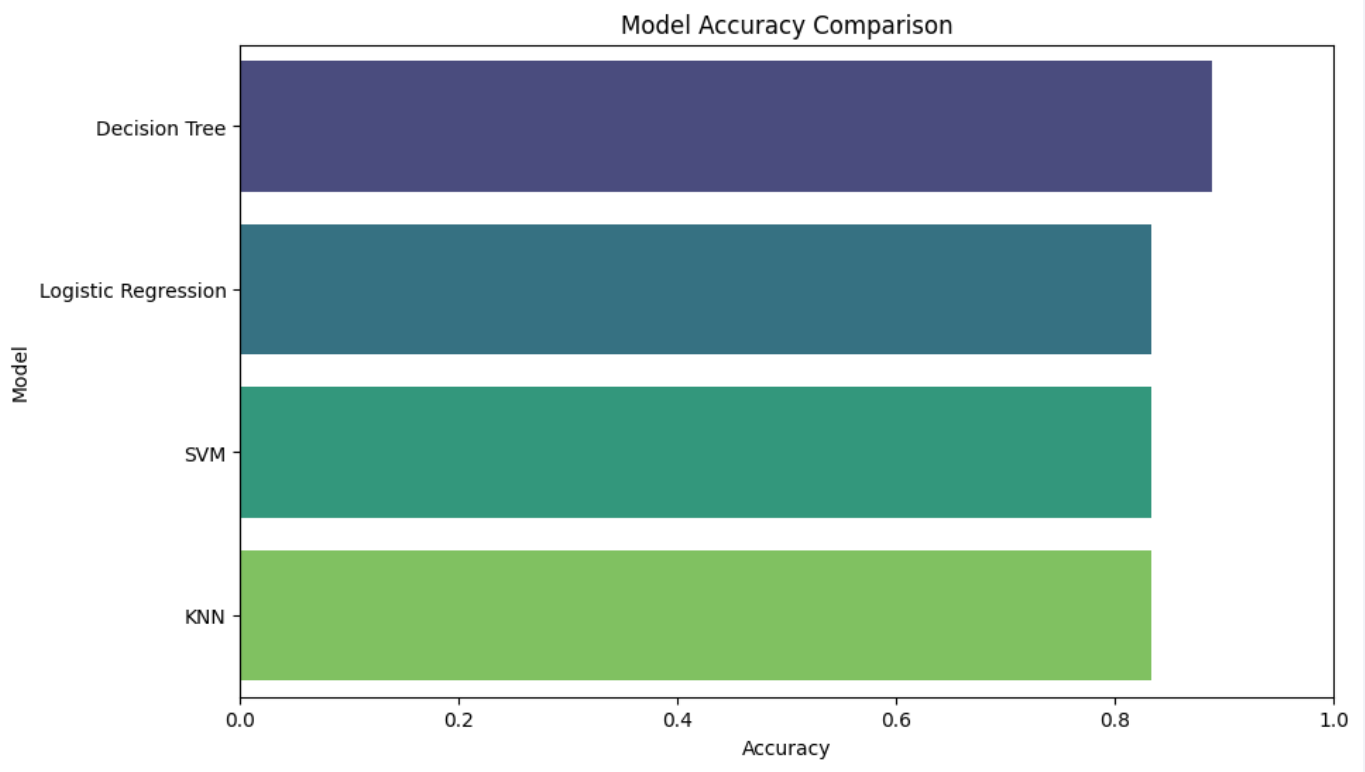
- Boster version FT is the most prevalent in (3000-8000)kg weight category
- Follwed by B4 and v1.1
- No successful launch of B5 in the above-mentioned payload

Section 5

Predictive Analysis (Classification)

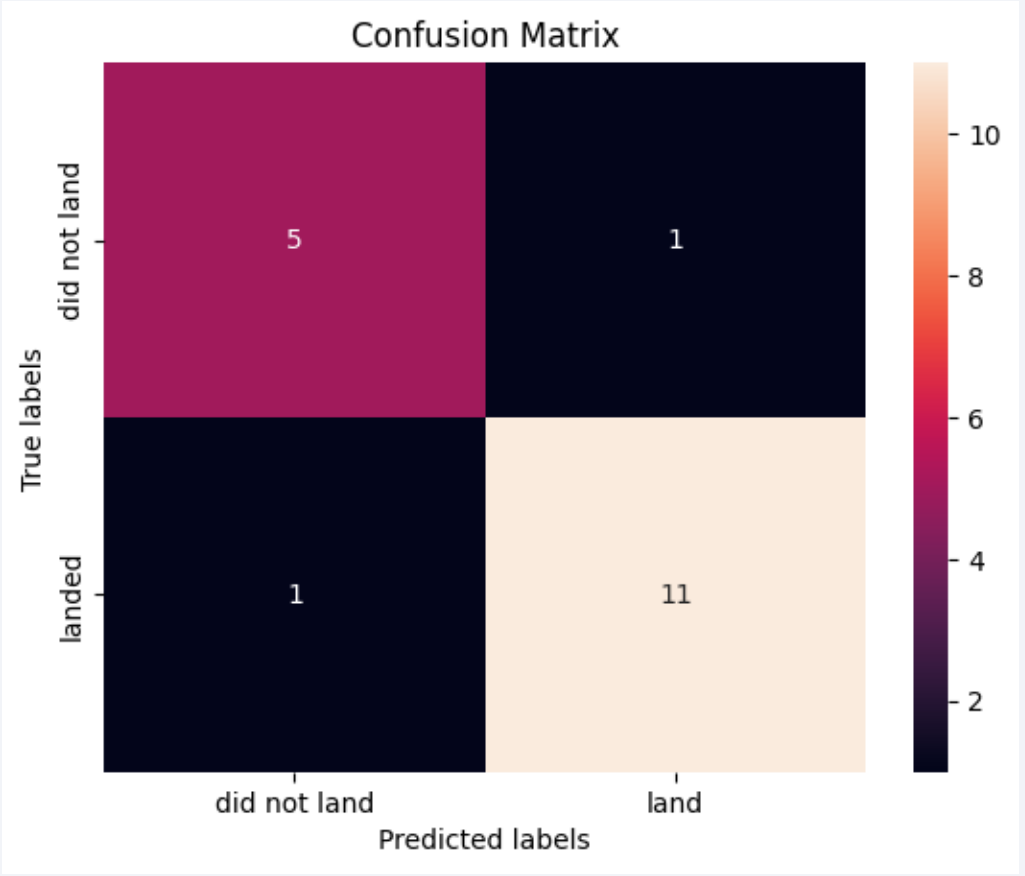
Classification Accuracy

- Decision Tree: 88.9%
- Logistic Regression: 83.3%
- SVM: 83.3%
- KNN: 83.3%



Confusion Matrix

- True Positives (11), model predicted rocket land when the rocket landed.
- False Positives (1), model wrongly predicted landing when it did not land.
- False Negatives (1), model predicted rocket did not landed when it landed.
- True Negatives (5), model predicted rocket did not land when it did not.



Conclusions

- The Decision Tree model emerged as the most effective classification method for predicting successful landings, with an accuracy of 88.9%.
- Launch success rates have steadily improved over time, with a marked increase after 2017 signifying SpaceX's learning curve and technological maturity.
- Launch site and orbit type significantly affect landing outcomes, with KSC LC-39A having the highest success rate and SO orbit showing zero success.
- Interactive dashboards and geospatial visualizations allowed for intuitive exploration of success rates based on launch site, payload, and orbital characteristics.
- The combination of API-based and web-scraped data provided a robust dataset, demonstrating how diverse data sources can enrich predictive modeling in aerospace.

Appendix

❖ Links of Project Materials

- Data Collection Using API
- Data Collection By Web Scraping
- Data Wrangling
- SQL Queries
- Data Visualization
- Launch Site Visualization Using Folium
- Dashboard
- Machine Learning Prediction

Thank you!