IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Jacques Nakkash
July 27, 2024

Outline

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

Executive Summary

Summary of Methodologies:

Data Collection: Data was primarily gathered from the SpaceX REST API using GET requests via the requests library.
Data Wrangling: Falcon 1 launches were excluded, missing values were imputed with the mean, and new features were engineered to capture factors influencing landing outcomes.
Exploratory Data Analysis (EDA): EDA was conducted using visualization tools and SQL for deeper insights.
Interactive Visual Analytics: Folium and Plotly Dash were used to create interactive visual analytics.
Predictive Analysis: The process involved preprocessing the data, splitting it into training and testing sets, training classification models, and evaluating the best model based on testing data.

Summary of All Results

The project successfully collected and normalized launch data from the SpaceX API and Wikipedia pages, wrangled and cleaned the data by filters and helper functions, queried the data for relevant information, successfully used the query results to plot charts and graphs and integrate them into interactive visuals such as dashboards and folium maps and finally predicted the best model to move forward with for predictive analysis

The Logistic Regression model achieved a notable test accuracy with a test accuracy of 83%

Introduction

Background

The SpaceX Falcon 9 first stage landing prediction project is aimed at leveraging historical launch data to predict whether the first stage of the Falcon 9 rocket will successfully land. SpaceX has revolutionized the space industry by developing reusable rocket technology, significantly reducing the cost of space missions. The ability to reuse the first stage of the Falcon 9 rocket is a key factor in these cost savings. Accurately predicting the landing outcome is crucial for planning and cost estimation of future missions. This project involves collecting and analyzing data from the SpaceX and related web sources to develop machine learning models that can predict landing success, providing valuable insights for SpaceX and potential competitors in the aerospace industry.

Problems We Want to Find Answers To

- Can we accurately predict whether the first stage of the Falcon 9 rocket will successfully land based on historical launch data?
- What are the key factors and features that most significantly influence the success or failure of the first stage landing?
- Which machine learning model (Logistic Regression, Support Vector Machine, Decision Tree, K-Nearest Neighbors) performs best in predicting the landing outcomes?
- How can we handle missing values and filter irrelevant data (such as Falcon 1 launches) to ensure a clean and meaningful dataset for analysis?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - API Endpoint Usage: Data was primarily collected from the SpaceX REST API, accessed through GET requests using the requests library.
- Perform data wrangling
  - Falcon 1 launches were filtered out, empty values were replaced with the mean, and new features were created to capture factors influencing landing outcomes
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Preprocessing data, splitting the data into training and testing data subsets, training the model, analyzing best model for analysis using testing data

Data Collection

Data Collection Process

Start Data Collection
•Initiate the data collection process.

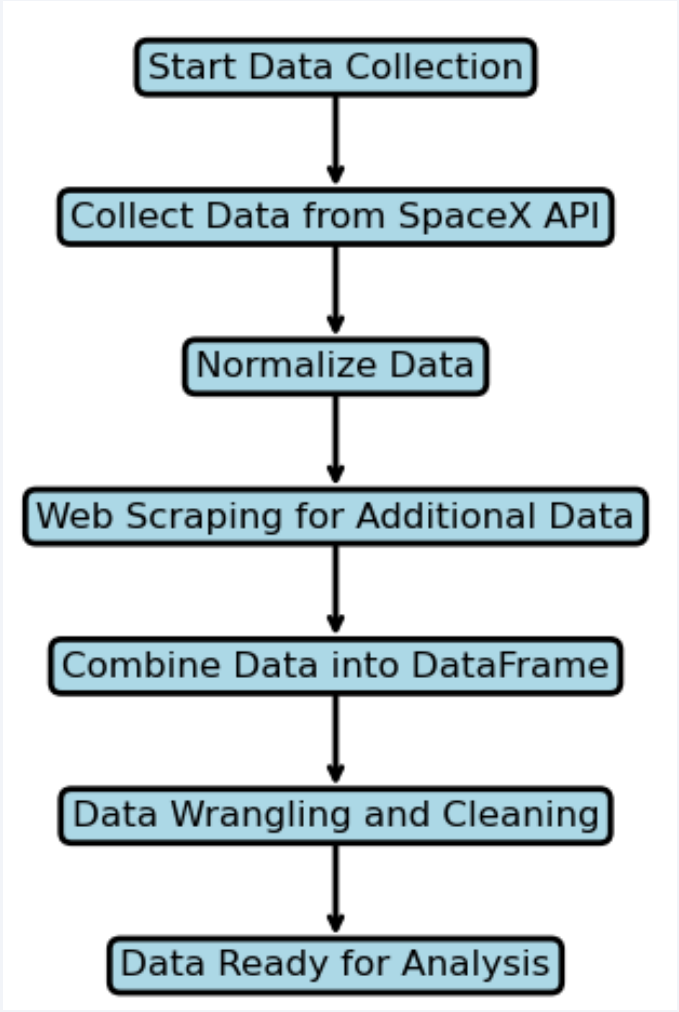
Collect Data from SpaceX API
•Fetch data using the SpaceX REST API for launches, rockets, launchpads, payloads, and cores.

Normalize Data
•Convert JSON responses into flat tables.

Web Scraping for Additional Data
•Scrape Wikipedia pages for additional Falcon 9 launch records.

Combine Data into DataFrame
•Integrate API data and web scraped data into a single data frame.

Data Wrangling and Cleaning
•Filter out Falcon 1 launches: filter empty values/create new features to clean the dataset for analysis



Data Collection – SpaceX API

API Endpoints and Calls

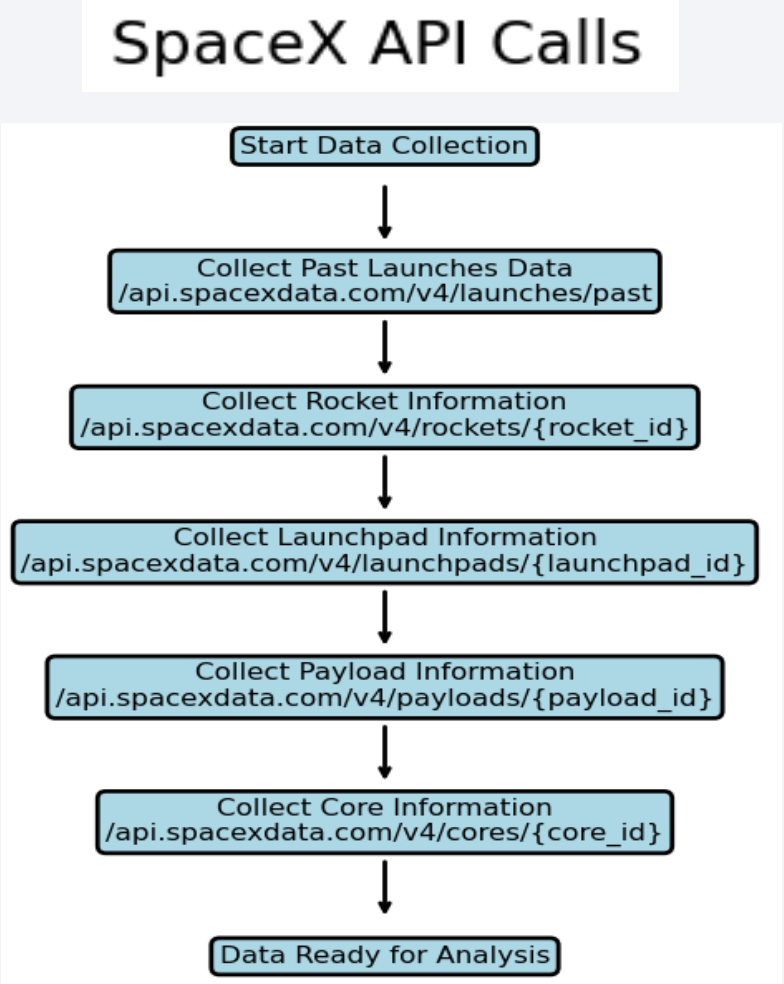
Collecting Past Launches:
•Endpoint: https://api.spacexdata.com/v4/launches/past
•This call retrieves detailed information about all past SpaceX launches.

Collecting Rocket Information:
•Endpoint: https://api.spacexdata.com/v4/rockets/{rocket_id}
•This call retrieves detailed information about a specific rocket using its unique rocket ID obtained from the launches data.

Collecting Launchpad Information:
•Endpoint: https://api.spacexdata.com/v4/launchpads/{launchpad_id}
•This call retrieves detailed information about a specific launchpad using its unique launchpad ID obtained from the launches data.

Collecting Payload Information:
•Endpoint: https://api.spacexdata.com/v4/payloads/{payload_id}
•This call retrieves detailed information about a specific payload using its unique payload ID obtained from the launches data.

Collecting Core Information:
•Endpoint: https://api.spacexdata.com/v4/cores/{core_id}
•This call retrieves detailed information about a specific core using its unique core ID obtained from the launches data.



Data Collection - Scraping

Steps to Perform Web Scraping

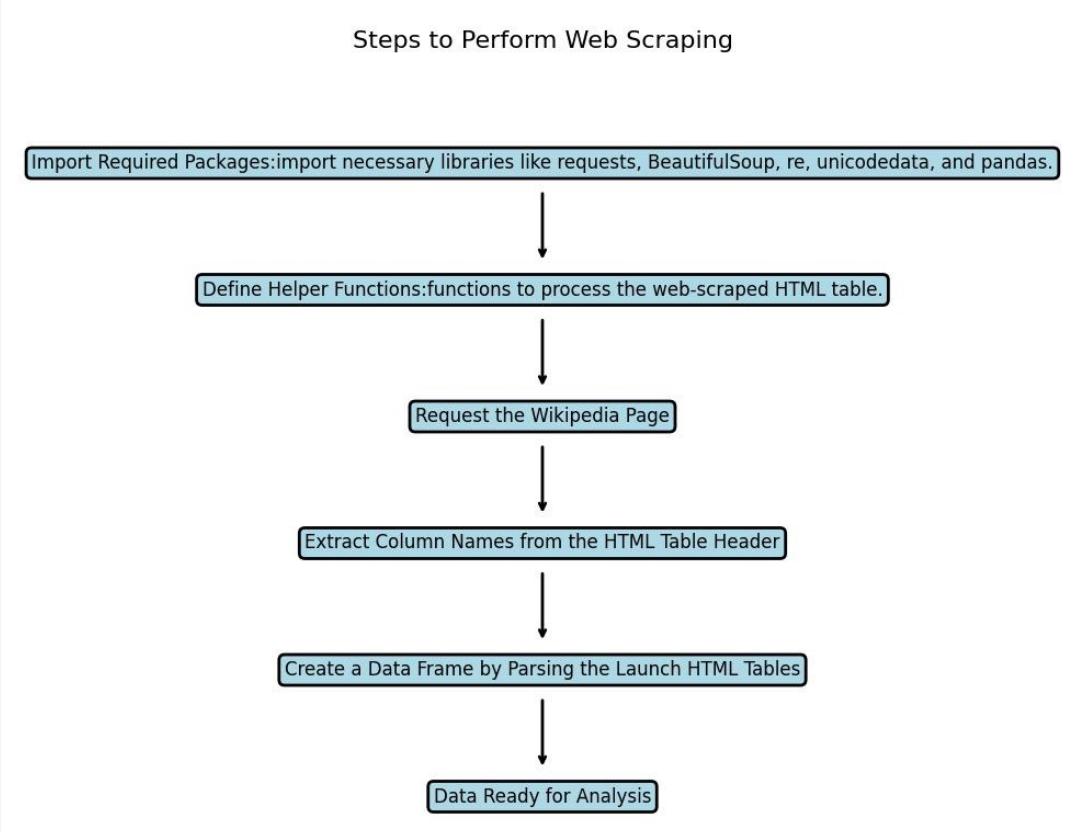
1.Import Required Packages:
Install and import necessary libraries

2.Define Helper Functions:
Define functions to process the web-scraped HTML table

3.Request the Wikipedia Page:
request the Falcon 9 Launch HTML page and create object from the response.

4.Extract Column Names from the HTML Table Header:
Find all tables on the Wikipedia page and extract column names).

5.Create a Data Frame by Parsing the Launch HTML Tables:
Initialize a dictionary with column names and iterate through the table rows to extract and fill in the launch record values.



Data Wrangling

Data Wrangling Process

1.Import Libraries: The necessary libraries for data manipulation and analysis were imported.

2.Load the Dataset: The SpaceX dataset was loaded from a CSV file into a Pandas DataFrame.

3.Initial Data Inspection: The first few rows of the Data Frame were displayed to understand the structure of the data.

4.Identify Missing Values: The percentage of missing values in each column was calculated to identify incomplete data.

5.Identify Data Types: The data types of each column were determined to differentiate between numerical and categorical data.

6.Count Launches by Site: The number of launches from each launch site was calculated to understand the distribution of launches.

7.Count Occurrences of Each Orbit: The occurrences of each orbit type were calculated to understand the distribution of orbits.

8.Count Mission Outcomes: The occurrences of each mission outcome were calculated to understand the success and failure rates.

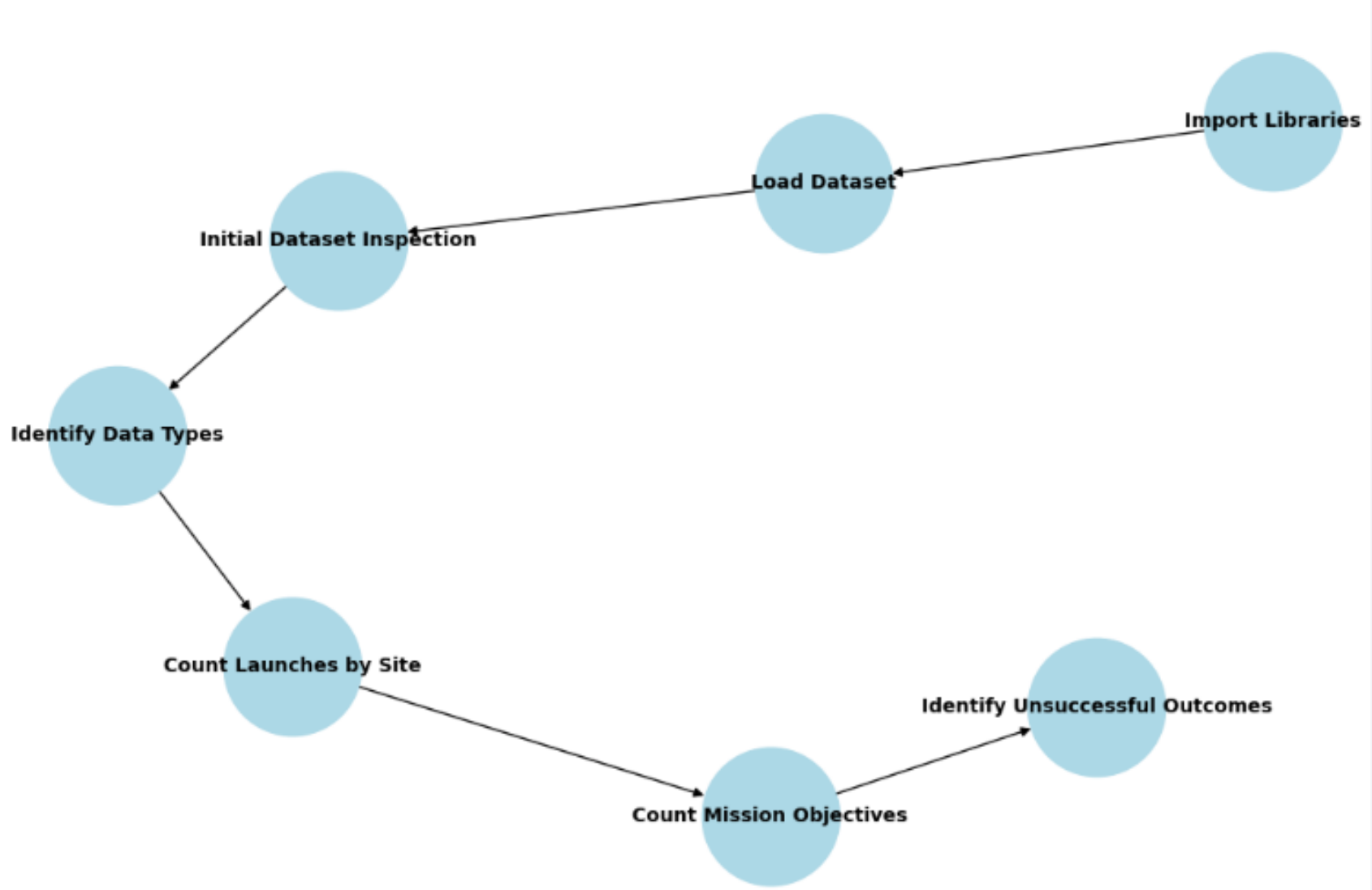
9.Identify Unsuccessful Outcomes: A set of outcomes where the second stage did not land successfully was created.

10.Create Landing Outcome Label: A new column, Class, was created to label successful (1) and unsuccessful (0) landings.

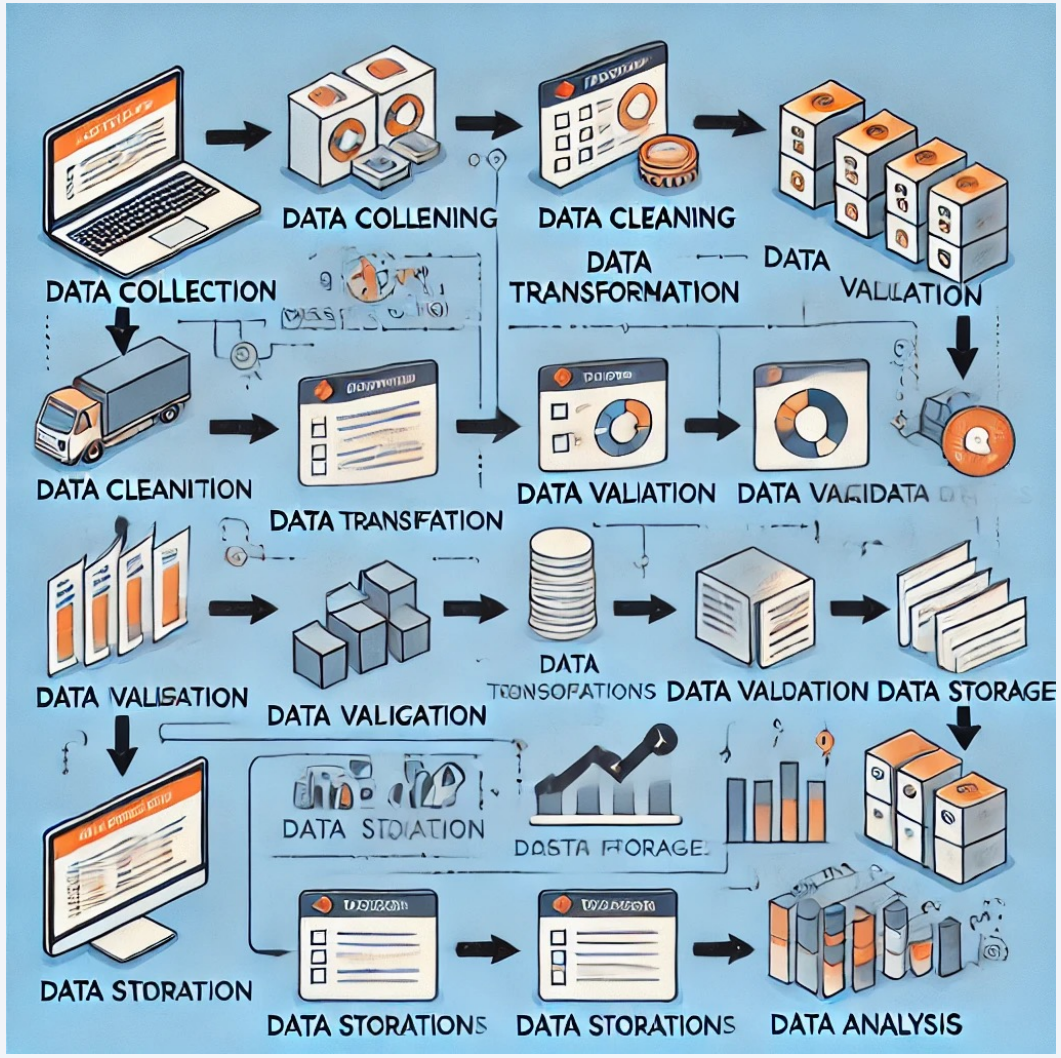
11.Calculate Success Rate: The success rate of the landings was calculated to determine the overall performance.

12.Export Processed Data: The processed Data Frame was exported to a CSV file for further analysis.

Data Wrangling: High level overview



Data Wrangling: Detailed Overview



EDA with Data Visualization

Charts Plotted

1.Scatter Plot: Flight Number vs. Payload Mass

Chart: A scatter plot with Flight Number on the x-axis, Payload Mass on the y-axis, and the hue representing the class value (success or failure).
Reason: This plot was used to visualize how the payload mass and the number of flight attempts correlate with the success of the launch. It helps to see if there is any trend in success rate as the flight number increases or as the payload mass changes.

2.Scatter Plot: Payload Mass vs. Launch Site

Chart: A scatter plot with Payload Mass on the x-axis, Launch Site on the y-axis, and the hue representing the class value.
Reason: This plot was used to observe if there is any relationship between launch sites and their payload mass. It helps to identify if certain launch sites handle heavier or lighter payloads and the success rate associated with them.

3.Bar Chart: Success Rate by Orbit Type

Chart: A bar chart showing the success rate for each orbit type.
Reason: This bar chart helps to visually check if there are any relationships between success rates and different orbit types. It allows for easy comparison of success rates across various orbit types.

4.Scatter Plot: Flight Number vs. Orbit Type

Chart: A scatter plot with Flight Number on the x-axis, Orbit Type on the y-axis, and the hue representing the class value.
Reason: This plot was used to see if there is any relationship between the flight number and orbit type. It helps to identify if certain orbits are more challenging at specific stages of the flight program.

5.Scatter Plot: Payload Mass vs. Orbit Type

Chart: A scatter plot with Payload Mass on the x-axis, Orbit Type on the y-axis, and the hue representing the class value.
Reason: This plot was used to visualize the relationship between payload mass and orbit type. It helps to see how different payload masses are distributed across different orbit types and their success rates.

6.Line Chart: Yearly Success Rate Trend

Chart: A line chart with the extracted year on the x-axis and the average success rate on the y-axis.
Reason: This chart was used to visualize the trend in launch success rates over the years. It helps to observe how the success rate has changed over time and if there are any noticeable improvements.

https://github.com/jnakkash/Capstone/blob/main/edadataviz%20(1).ipynb

EDA with SQL

SQL Queries Performed

- Query of unique launch site names.
- Query of records based on launch site name.
- Query of total payload mass for specific missions.
- Query of average payload mass for a specific booster version.
- Query of the first successful landing date on a ground pad.
- Query of boosters with specific success criteria and payload mass range.
- Query of successful and failed mission outcomes.
- Query using subqueries to find booster versions with maximum payload mass.
- Query of records with specific criteria for failure outcomes, booster versions, and launch sites.
- Query of landing outcomes ranked within a specific date range.

https://github.com/jnakkash/Capstone/blob/main/EDA_SQL_Queries.ipynb

Build an Interactive Map with Folium

Map Objects Added to the Folium Map

1.Markers:

Launch Records Markers: Green markers for successful launches and red markers for failed launches

Launch Site Labels: markers to label each launch site with its name.

Coastline Distance Marker: A marker at the closest coastline point displaying the calculated distance from the launch site.

2.Circles:

Launch Site Highlight Circles: Circles around each launch site to highlight their locations.

3.Lines:

Distance Line: A blue polyline between the launch site and the closest coastline point to visually represent the calculated distance.

4.Mouse Position:

Mouse Position Plugin: To display the latitude and longitude of the mouse pointer's current position on the map.

https://github.com/jnakkash/Capstone/blob/main/Folium%20Map.ipynb

Build a Dashboard with Plotly Dash

Plots/Graphs and Interactions Added to the Dashboard

- **Dropdown List**:
  - **Purpose**: Allows users to select a specific launch site or view data for all sites.
  - **Interaction**: Dropdown menu updates the displayed data based on user selection.

- **Pie Chart**:
  - **Purpose**: Visualizes the success and failure counts of launches.
  - **Interaction**: Updates dynamically based on the selected launch site from the dropdown.

- **Range Slider**:
  - **Purpose**: Enables users to filter the data based on payload mass ranges.
  - **Interaction**: Users can adjust the range to update the scatter plot accordingly.

- **Scatter Plot**:
  - **Purpose**: Displays the relationship between payload mass and launch outcomes.
  - **Interaction**: Scatter plot updates based on selected launch site and payload range, with color coding for booster versions.

https://github.com/jnakkash/Capstone/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

Model Development Process

1.Data Processing

Load Data: Read the dataset into a Pandas DataFrame.
Feature Selection: Select relevant features and target variable.
Standardization: Standardize the features using StandardScaler.

2.Model Training and Hyperparameter Tuning

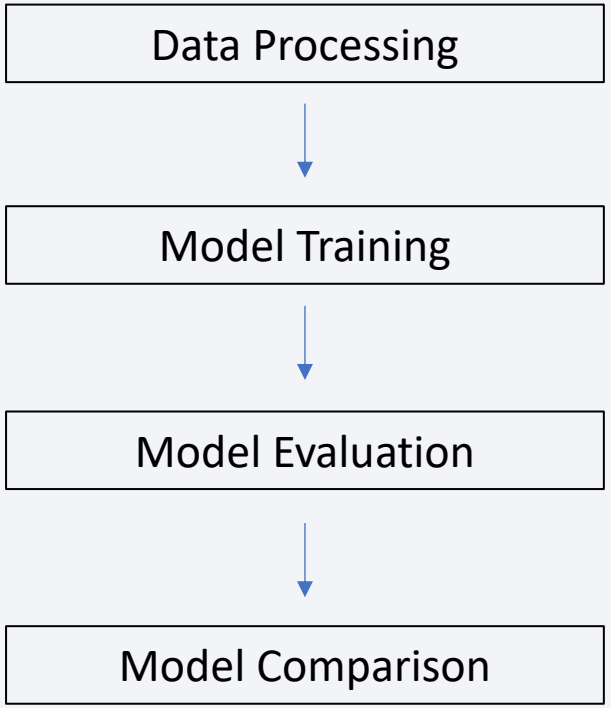
Logistic Regression:
Support Vector Machine (SVM):
Decision Tree:
K-Nearest Neighbors (KNN):

3.Model Evaluation

Accuracy Calculation: Calculate the accuracy of each model on the test data using the score method.
Confusion Matrix: Plot confusion matrix to visualize the performance of each model.

4.Model Comparison

Compare Models: Compare the test accuracies of all models.
Select Best Model: Identify the model with the highest test accuracy.



https://github.com/jnakkash/Capstone/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5%20(3).ipynb

Results Summary

Model Insights

The equal performance of Logistic Regression, SVM, and KNN suggests that the relationship between the features and the landing outcome might be relatively simple and well-captured by both linear (Logistic Regression) and non-linear (SVM, KNN) models.

Despite the same accuracy, the **Logistic Regression** model would be the most fit to move forward with due to its simplicity, fast training, less prone to overfitting and easy regulation

Model	Accuracy	Key Advantages
Logistic Regression	0.83	Simple, interpretable, fast training, less prone to overfitting, handles multicollinearity well, provides probabilistic outputs, easily regularized
SVM	0.83	Effective for high-dimensional spaces, robust to overfitting with proper kernel and regularization
Decision Tree	0.72	Easy to interpret, visualizable, handles both numerical and categorical data
K-Nearest Neighbors	0.83	Simple, effective in low-dimensional spaces, non-parametric

https://github.com/jnakkash/Capstone/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5%20(3).ipynb

Results: Interactive Demo

Interactive Analytics Demo using Streamlit

Model Performance Comparison

	Model	Accuracy
0	Logistic Regression	0.830000
1	SVM	0.830000
2	Decision Tree	0.720000
3	K-Nearest Neighbors	0.830000

Accuracy Comparison Across Models

Model Insights

Select a model to see insights:

SVM

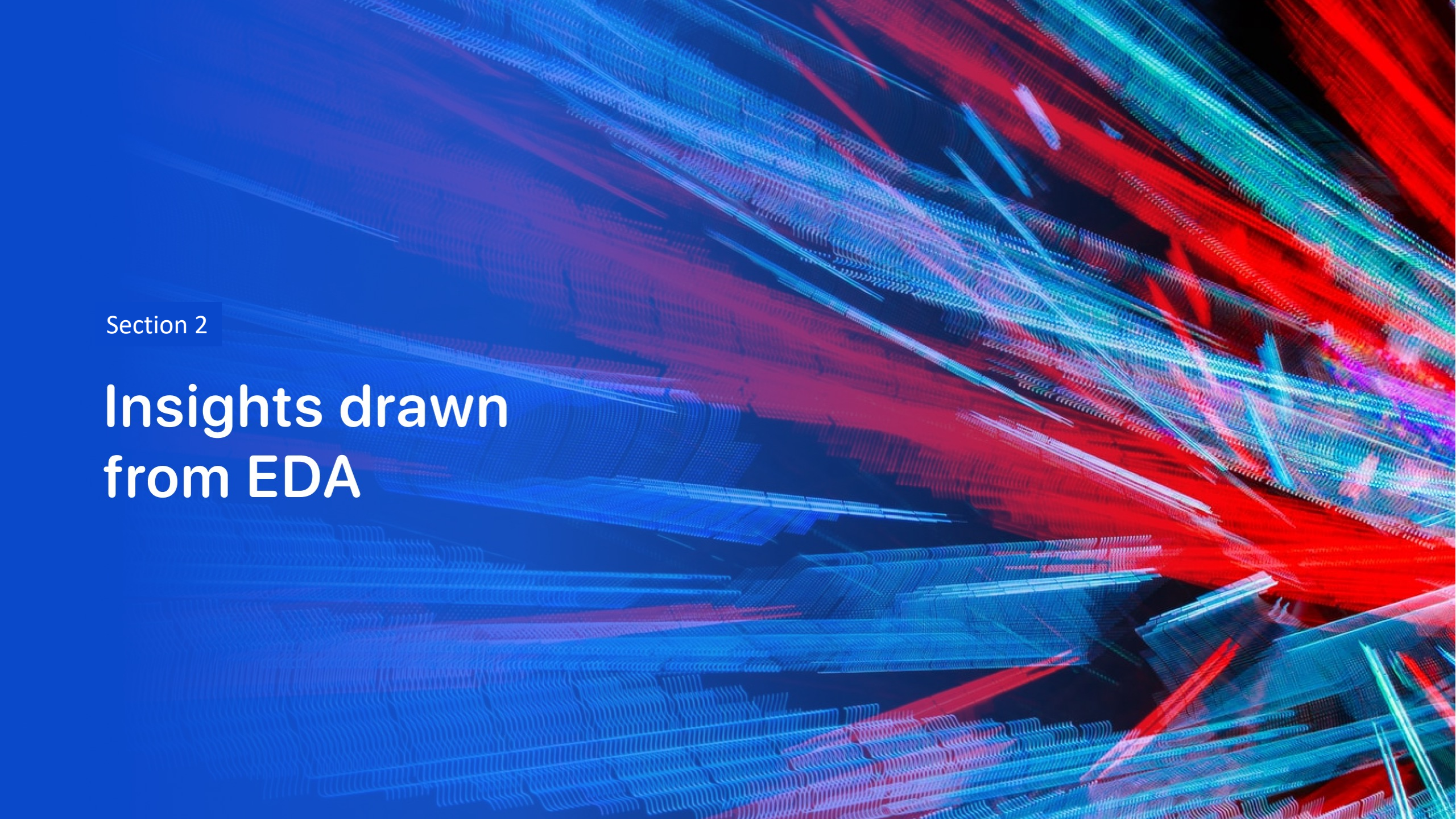
- Logistic Regression
- SVM
- Decision Tree
- K-Nearest Neighbors

Model Insights

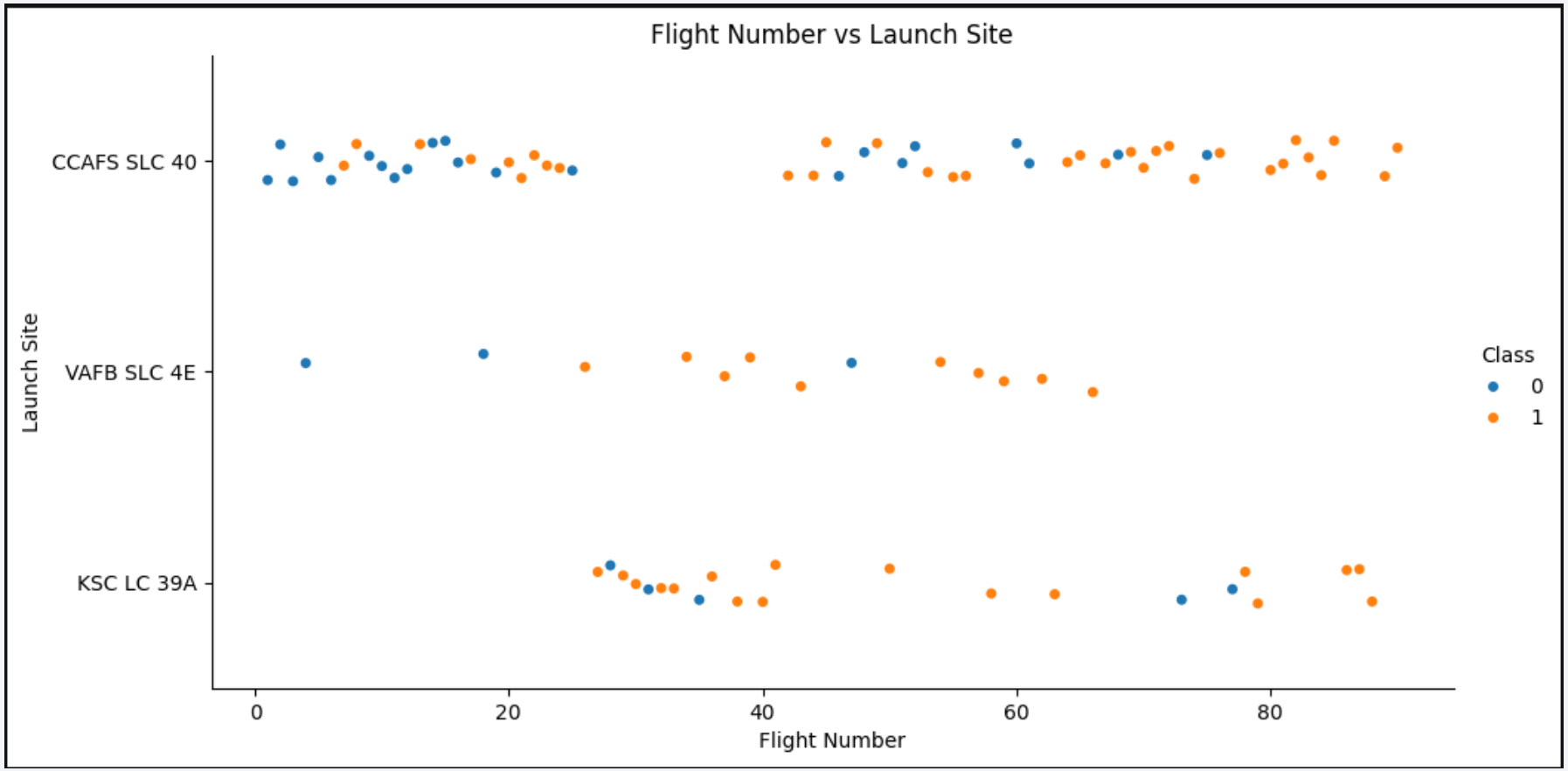
Select a model to see insights:

Decision Tree

Underperformed compared to other models with 0.72 accuracy. This suggests that simple threshold-based rules may not be sufficient for this prediction task. However, it offers high interpretability.

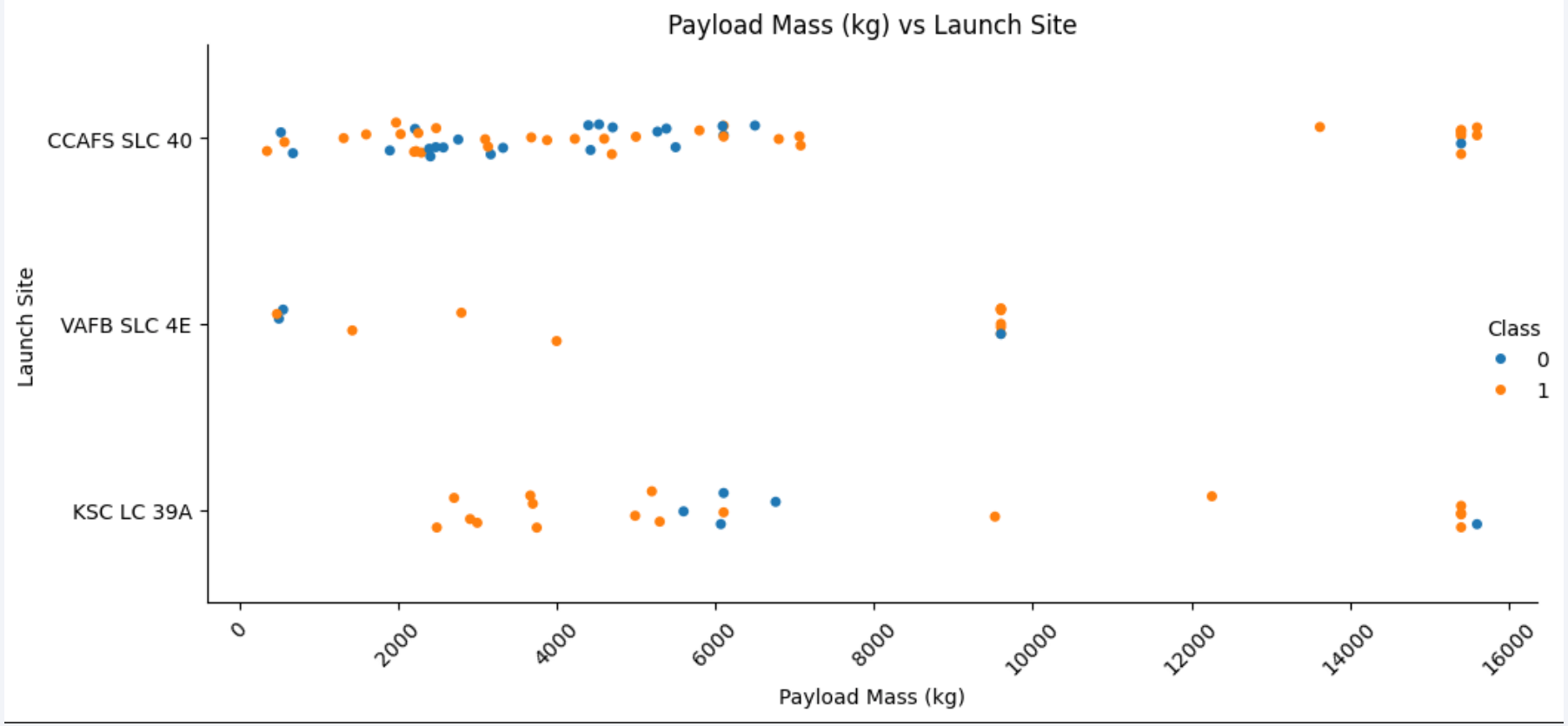

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site



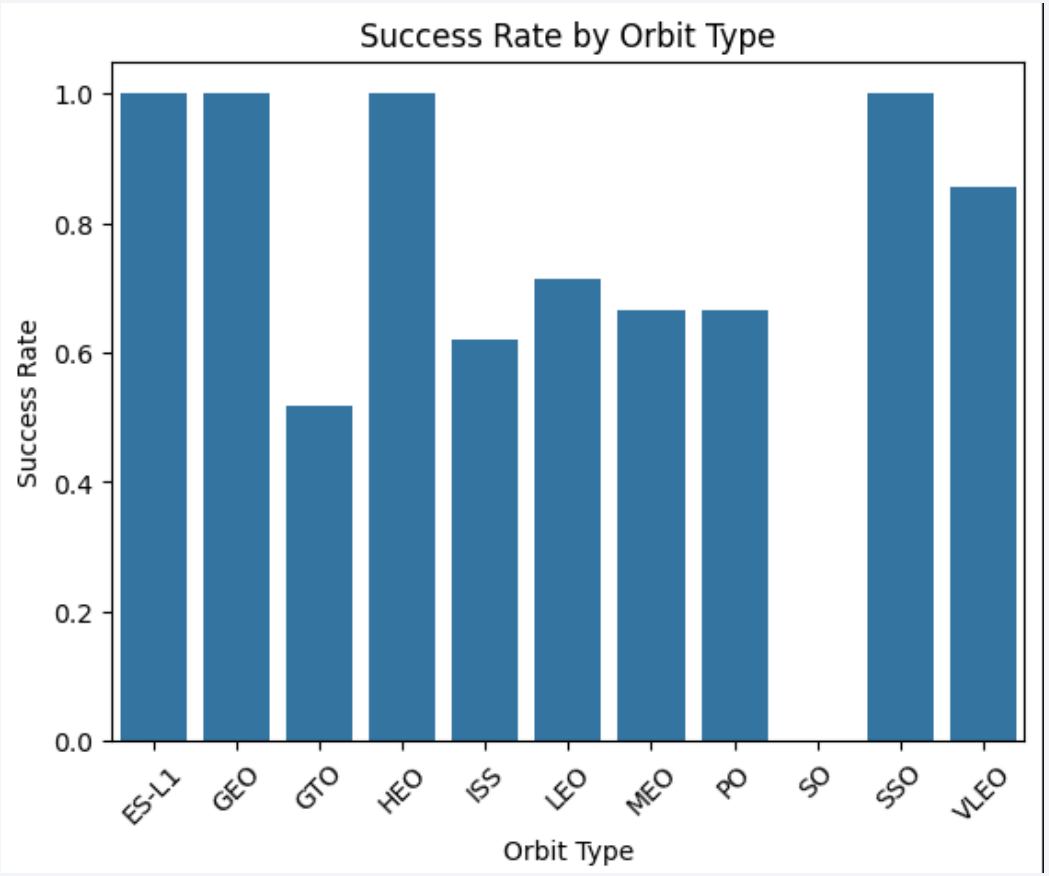
This chart helps identify different flight numbers at each launch site and identify any trends in success rate or failure as the flight number increases per launch site

Payload vs. Launch Site



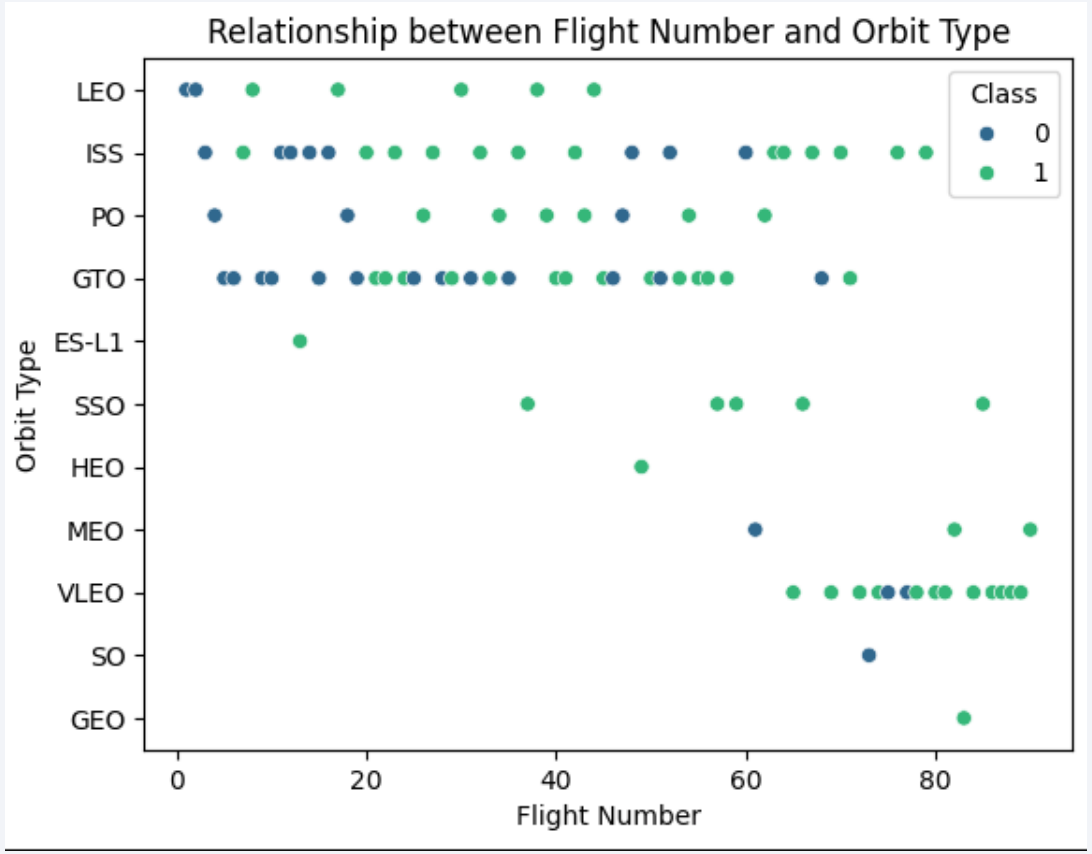
This plot was used to visualize how the payload mass and the launch site correlate with the success of the launch. It helps to see if there is any trend in success rate as the payload mass changes per launch site

Success Rate vs. Orbit Type



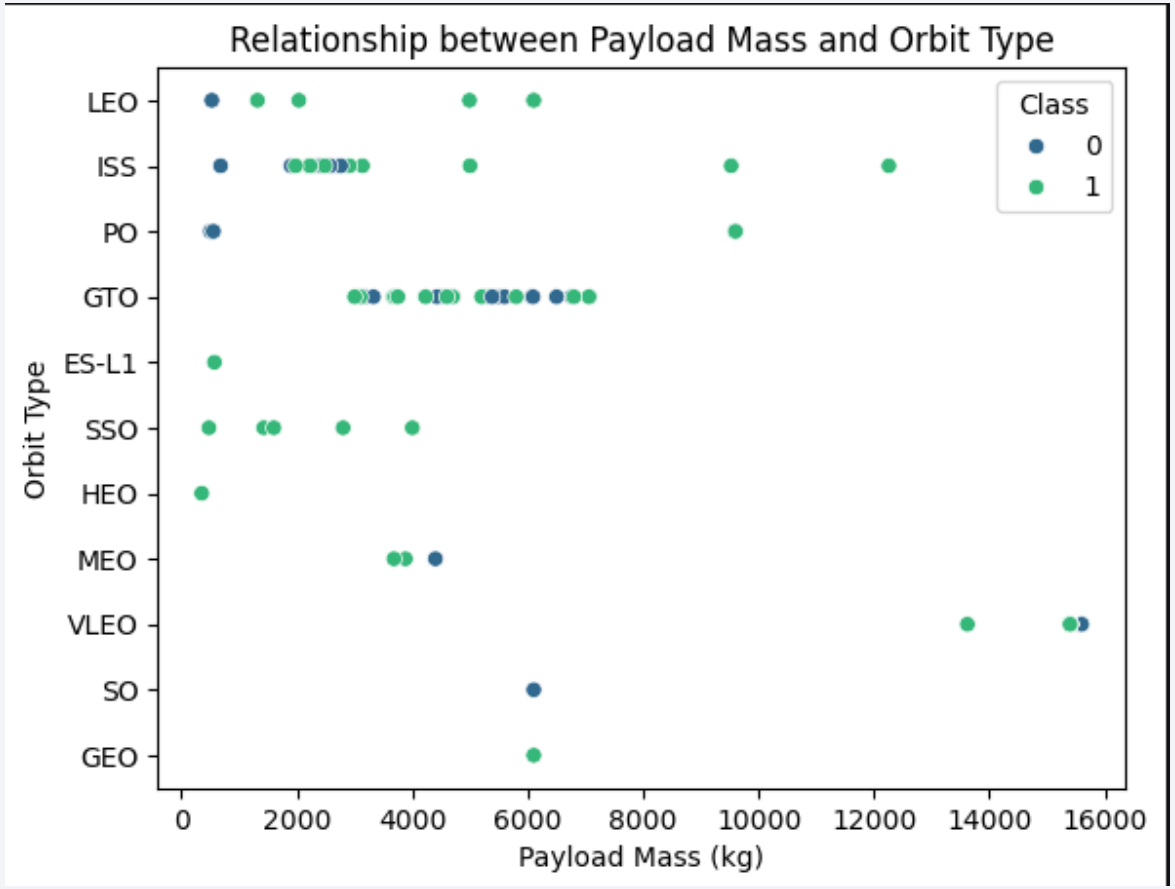
This bar chart helps to visually check if there are any relationships between success rates and different orbit types. It allows for easy comparison of success rates across various orbit types.

Flight Number vs. Orbit Type



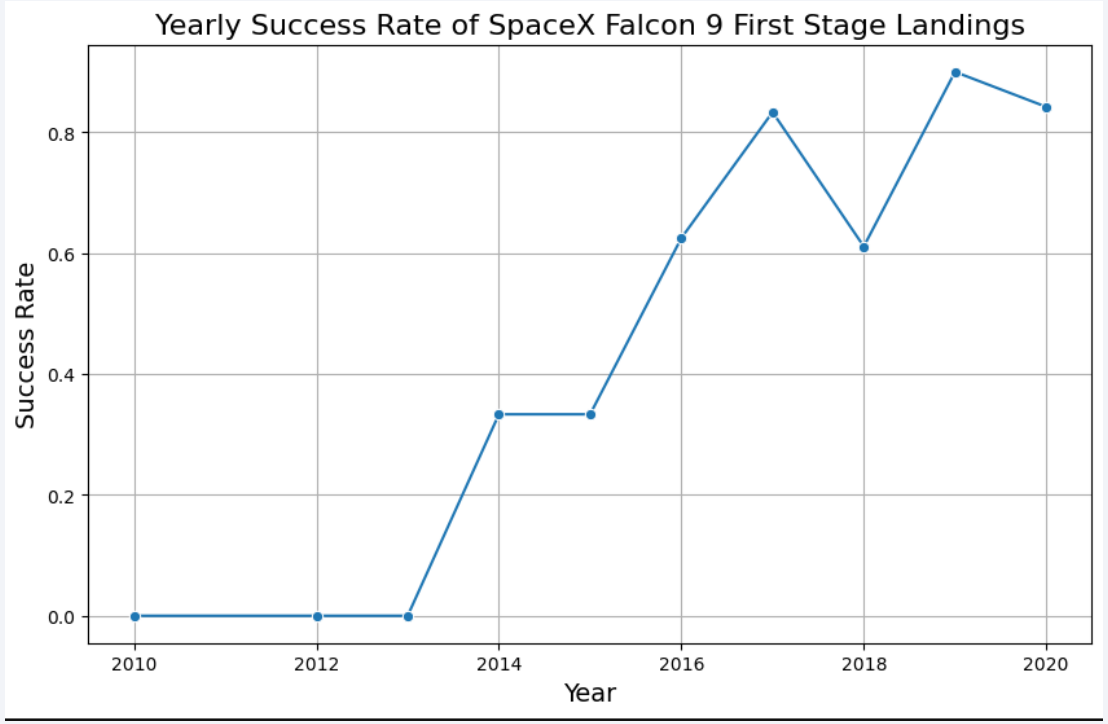
This plot was used to see if there is any relationship between the flight number and orbit type success rate. It helps to identify if certain orbits are more challenging at specific stages of the flight program

Payload vs. Orbit Type



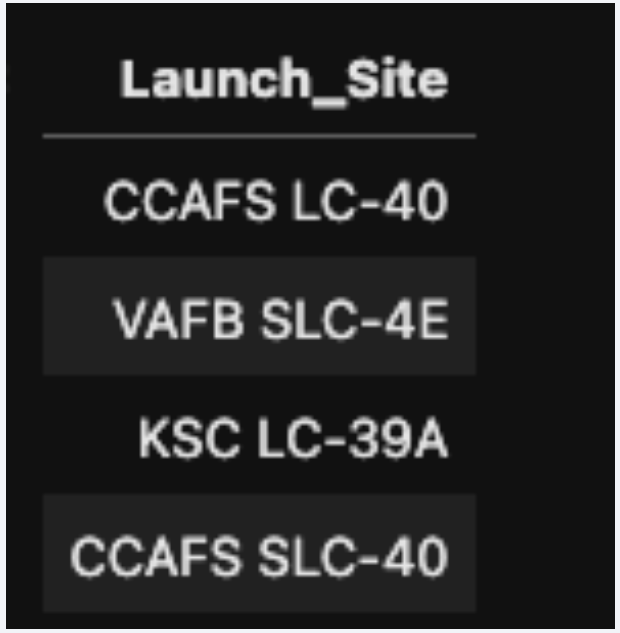
This scatterplot helps to visualizing increasing payload masses per each orbit type and if there any trends in success rates as the payload mass change for each orbit type

Launch Success Yearly Trend



This chart was used to visualize the trend in launch success rates over the years. It helps to observe how the success rate has changed over time and if there are any noticeable improvements

All Launch Site Names



Query of all unique Launch site names

Launch Site Names Begin with 'CCA'

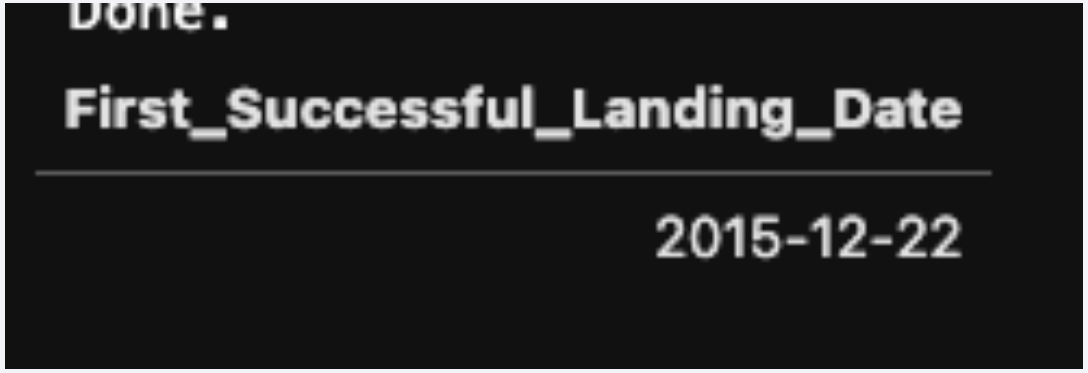
Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight 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

Query of 5 launch site names beginning with ‘CCA’

Average Payload Mass by F9 v1.1

Query of average payload mass for a specific booster version

First Successful Ground Landing Date



Query to find the first successful landing outcome on ground pad

Successful Drone Ship Landing with Payload between 4000 and 6000

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

Query of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

Total Number of Successful and Failure Mission Outcomes

Landing_Outcome	Count
Controlled (ocean)	5
Failure	3
Failure (drone ship)	5
Failure (parachute)	2
No attempt	21
No attempt	1
Precluded (drone ship)	1
Success	38
Success (drone ship)	14
Success (ground pad)	9
Uncontrolled (ocean)	2

The total number of successful and failure mission outcomes

Boosters Carried Maximum Payload

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

Query of the names of the booster which have carried the maximum payload type

2015 Launch Records

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

the failed landing outcomes, including mission information such as: month, booster versions, and launch site names for the year 2015

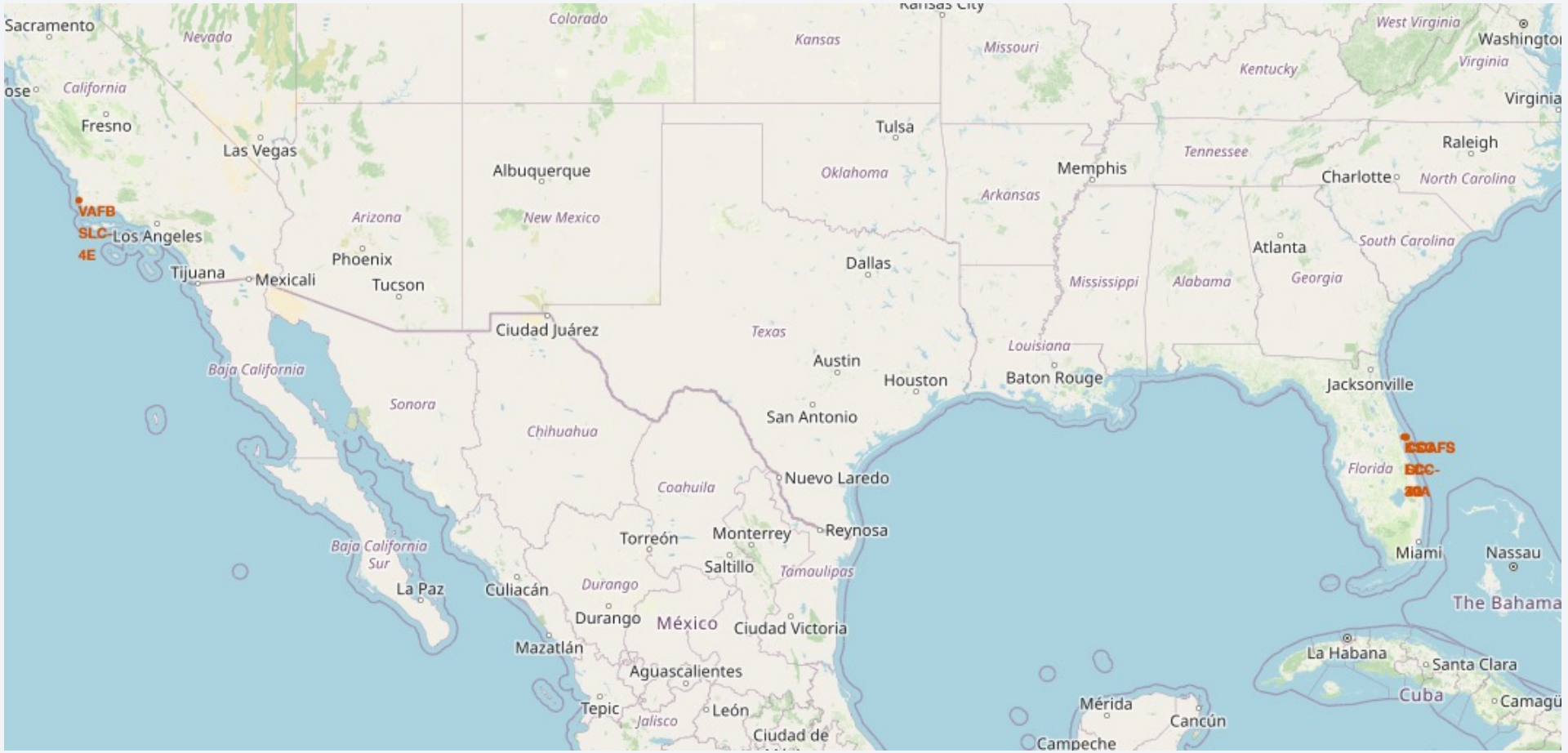
Rank Landing Outcomes Between 2010-06-04 and 2017-03-20

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

Flight landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order

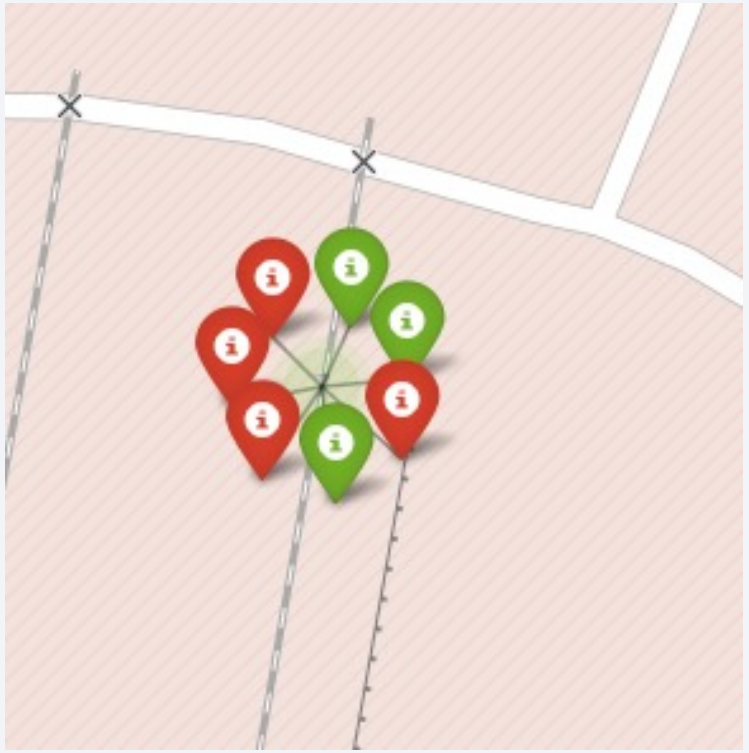

Section 3
Launch Sites
Proximities Analysis

Launch Sites



all launch sites' location markers on a global map

Site Launch Outcomes

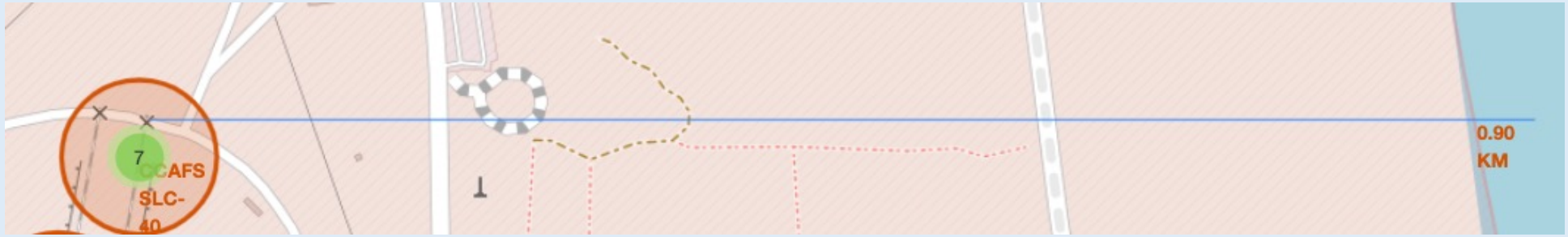


Site Launch Outcomes:

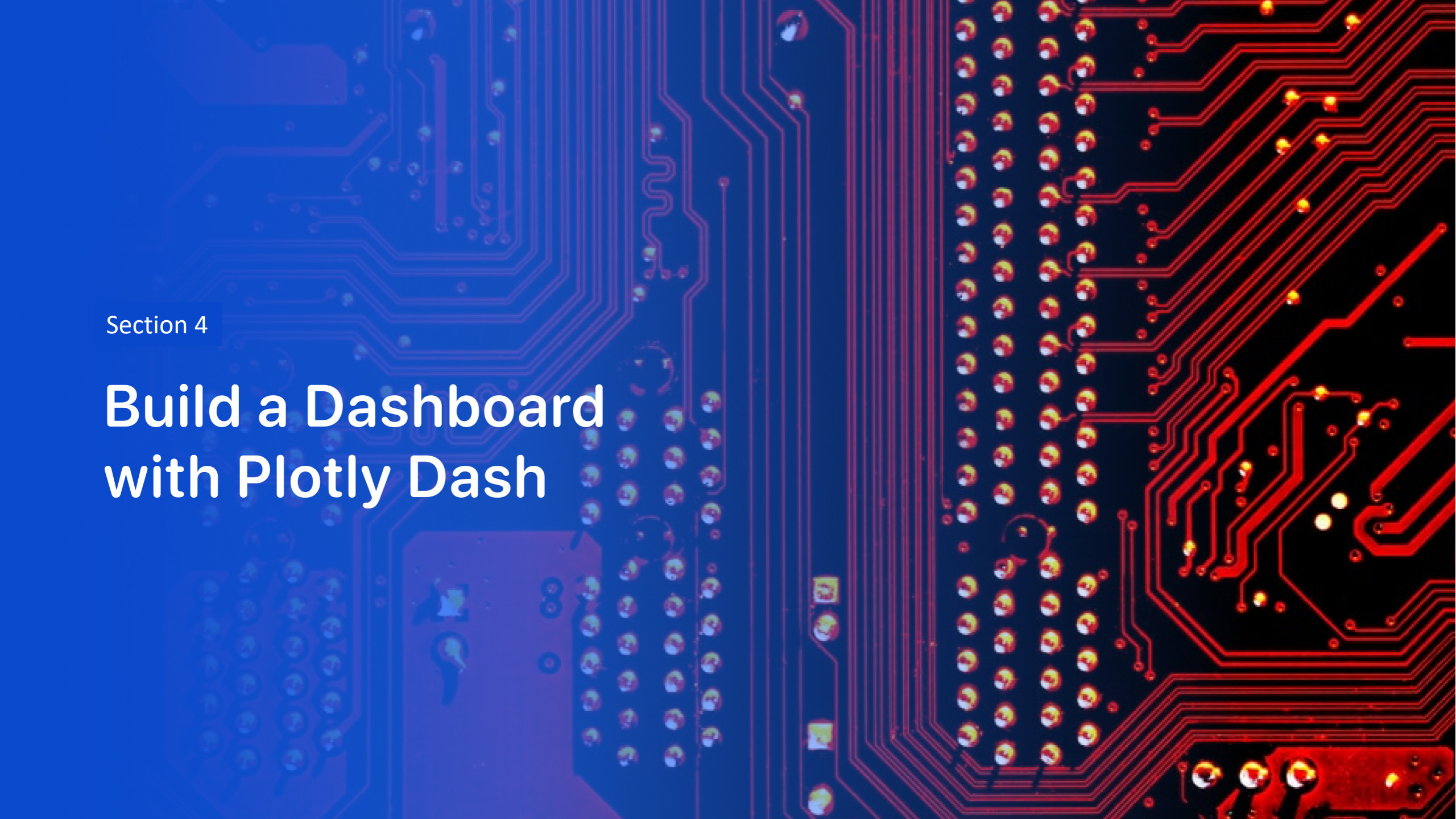
Successful launches marked in green

Failed launches marked in red

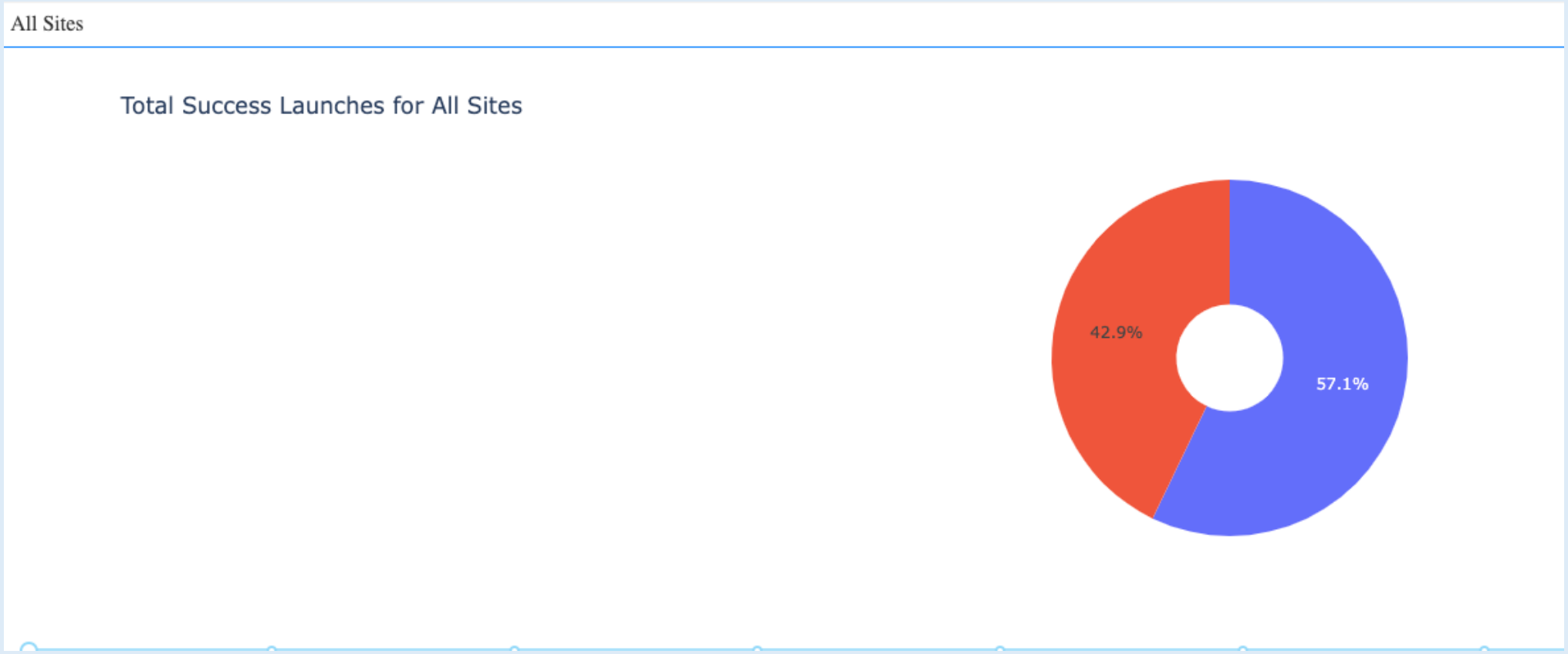
Launch Site Proximity Analysis



Proximity analysis of launch site to nearest coastline

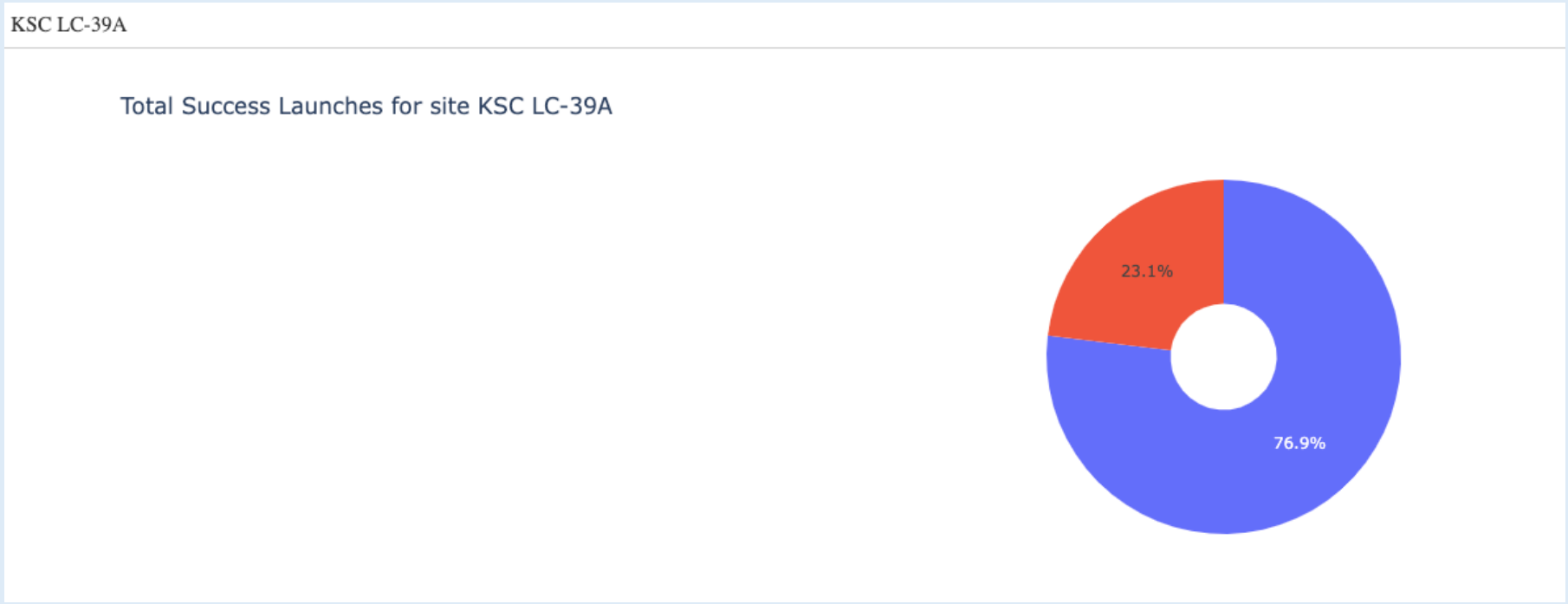

Section 4
Build a Dashboard with Plotly Dash

Launch Success Count



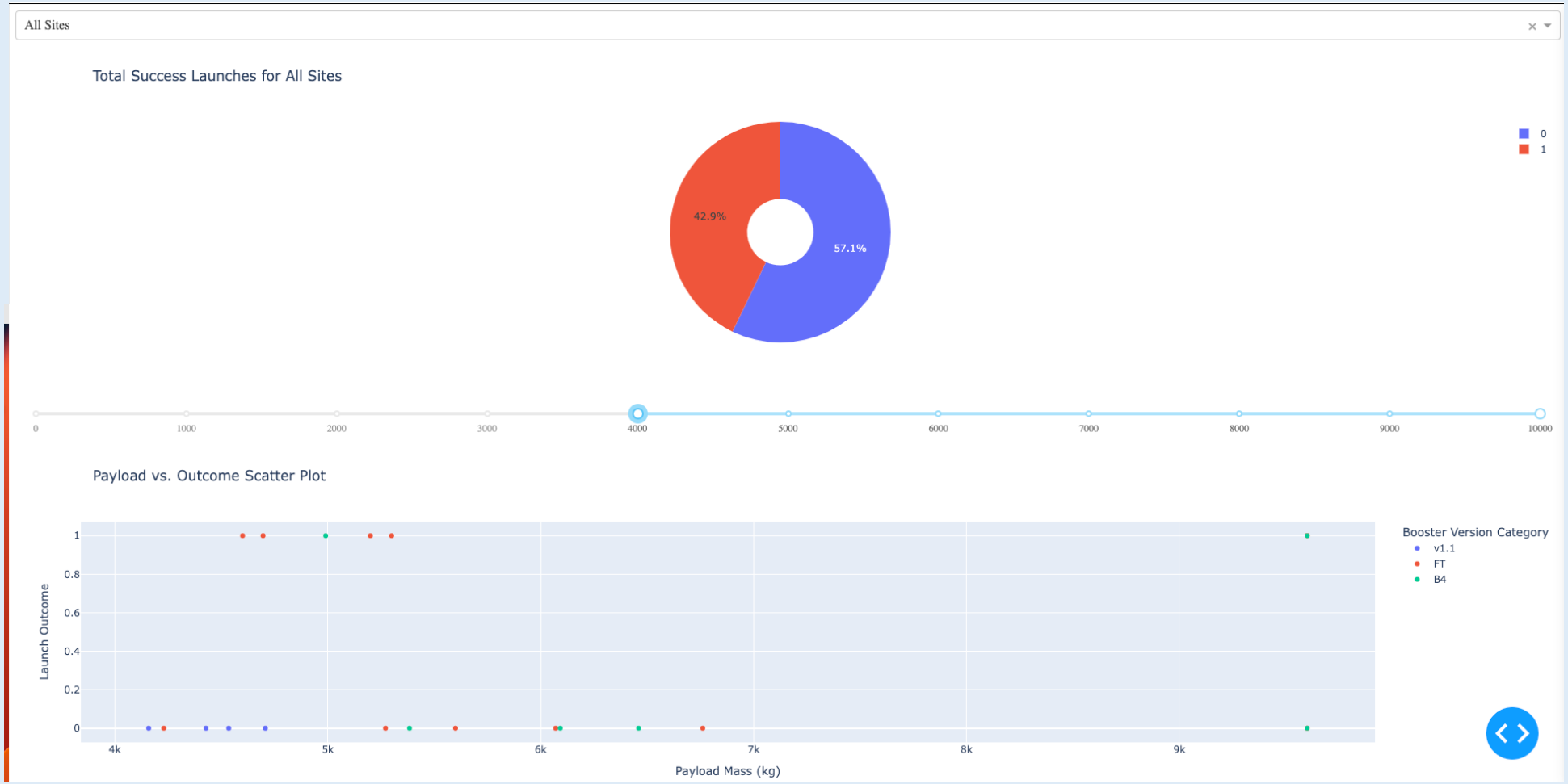
launch success count for all sites

Highest Success Ratio



KSC LC-39A: launch site with highest launch success ratio

Site Payload vs. Launch Outcome



4000kg Payload Launch Outcome scatter plot for all sites,

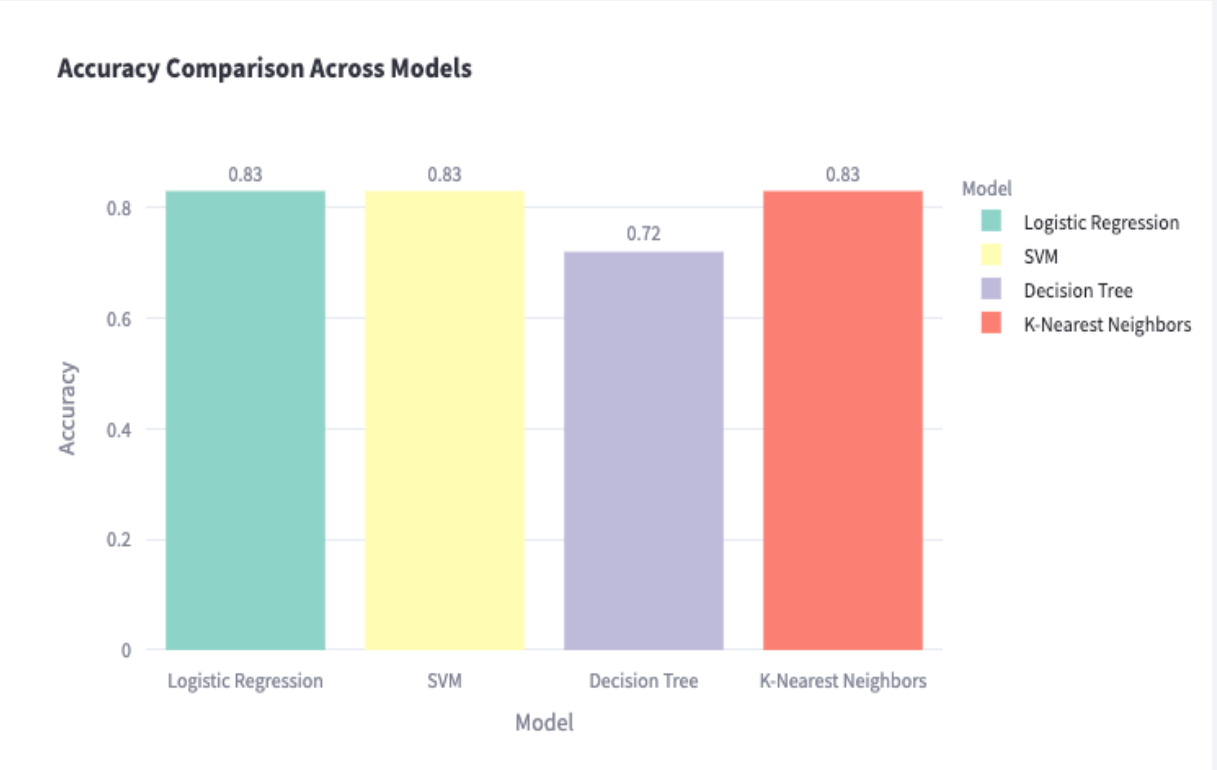
Section 5

Predictive Analysis (Classification)

Classification Accuracy

All three models, Logistic Regression, SVM, and K-Nearest Neighbors, have the same accuracy of **0.83**

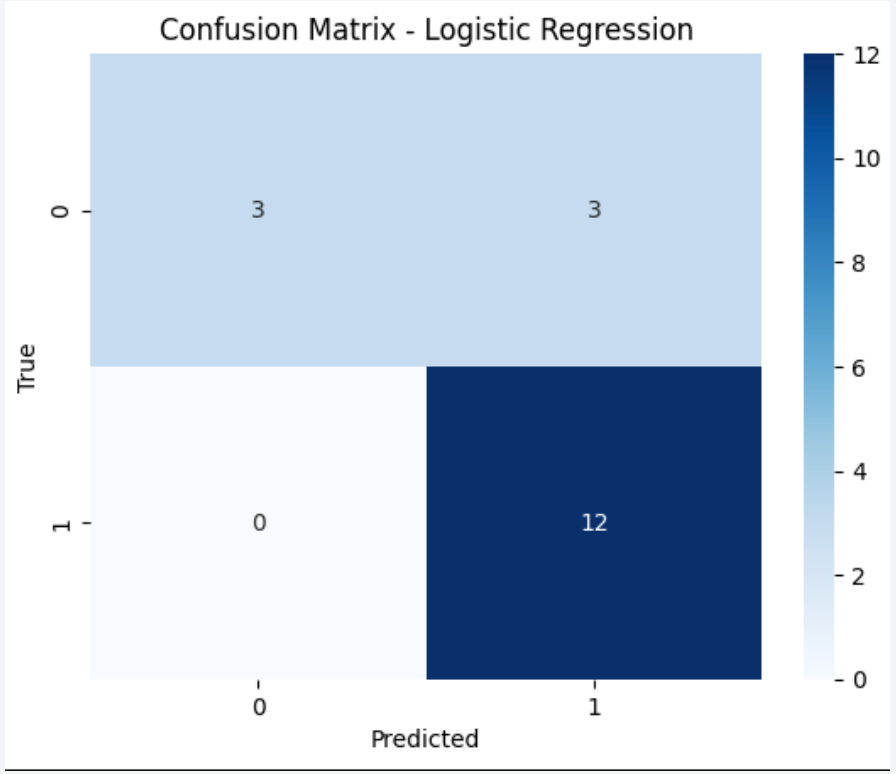
Model	Accuracy
Logistic Regression	0.83
SVM	0.83
Decision Tree	0.72
K-Nearest Neighbors	0.83



Confusion Matrix

The confusion matrix for the Logistic Regression model shows that it effectively identifies true positives and true negatives, though it makes some errors with false positives. With an accuracy of 83%, the model demonstrates good overall performance. It correctly predicted for all positive instances, indicating high reliability.

Despite some false positives, the model's ability to identify all actual positives instances makes it a strong choice for tasks where ensuring no positive cases are missed is crucial



Values in the Confusion Matrix:

- **True Negatives (TN):** 3 (Top-left cell)
- **False Positives (FP):** 3 (Top-right cell)
- **False Negatives (FN):** 0 (Bottom-left cell)
- **True Positives (TP):** 12 (Bottom-right cell)

Conclusions

- Logistic Regression Model is likely to be the best model due to simplicity, training speed and less error prone
- Reference the tables below to highlight the fact that even though the decision tree model performed the best while training, when It was used with actual data set, it performed the worst
- You can't always just rely on model training to fit your actual data

Model	Training Accuracy (using training data)
Logistic Regression	0.846
SVM	0.846
Decision Tree	0.875
K-Nearest Neighbors	0.848

vs

Model	Accuracy (Actual Data Set)
Logistic Regression	0.83
SVM	0.83
Decision Tree	0.72
K-Nearest Neighbors	0.83