Winning the Space Race with Data Science

Jack Conrick
November 2023

REPORT OUTLINE

Executive Summary

This project strategically positions a theoretical company as a competitive force in the dynamic space launch industry, leveraging machine learning for actionable insights. The report goes through key aspects of the methodology used in the project , including:

- **Data Collection and Wrangling**: Using Python, Jupyter notebooks, Pandas, NumPy, and BeautifulSoup for comprehensive data collection, cleaning, and feature engineering following API calls and web scraping, setting a robust foundation for subsequent analyses.
- **EDA and Visualization**: Employing SQL/SQLalchemy, visualizations, Folium, and Plotly Dash for exploring SpaceX's launch data trends and success factors.
- **Predictive Analysis**: Developing and validating machine learning models (logistic regression, SVM, Decision Tree, and KNN) using all mentioned libraries plus sklearn, including optimising hyperparameters using GridSearch.

Key findings and strategic insights include:

- Identifying SpaceX's consistent success improvement, highlighting KSC LC-39A as the most successful launch site. Discovering opportunities for targeted pricing strategies based on product and location, and increasing prediction precision.
- Selecting a K Nearest Neighbour machine learning model as the best approach for predicting success of a SpaceX launch.
- Equipping the company with actionable insights for cost estimation, competitive bidding, risk management, market differentiation, strategic planning, resource optimization, customer satisfaction, adaptability to market trends, and regulatory compliance, enhancing its competitive position in the industry.

Introduction

In the rapidly evolving landscape of space launch technologies, strategic decision-making is paramount for companies seeking a competitive edge.

This report delves into the use of machine learning to support predictive analysis of Falcon 9 first stage landings, utilizing AI to not only estimate launch success but also to empower businesses with cost-effective bidding strategies.

The report explores the distribution of SpaceX's market offerings against different orbit and payload types, as well as launch location, presenting opportunities for targeted pricing strategies and market differentiation. Importantly, it goes into detail on the technical steps required to apply machine learning to business decision making.

Through a data-driven approach, this report provides actionable insights and opportunities for further application of machine learning analysis, enabling companies to navigate the dynamic space launch industry with best-of-class AI processes to inform their strategic planning.

Section One

Methodology

Methodology

Executive Summary

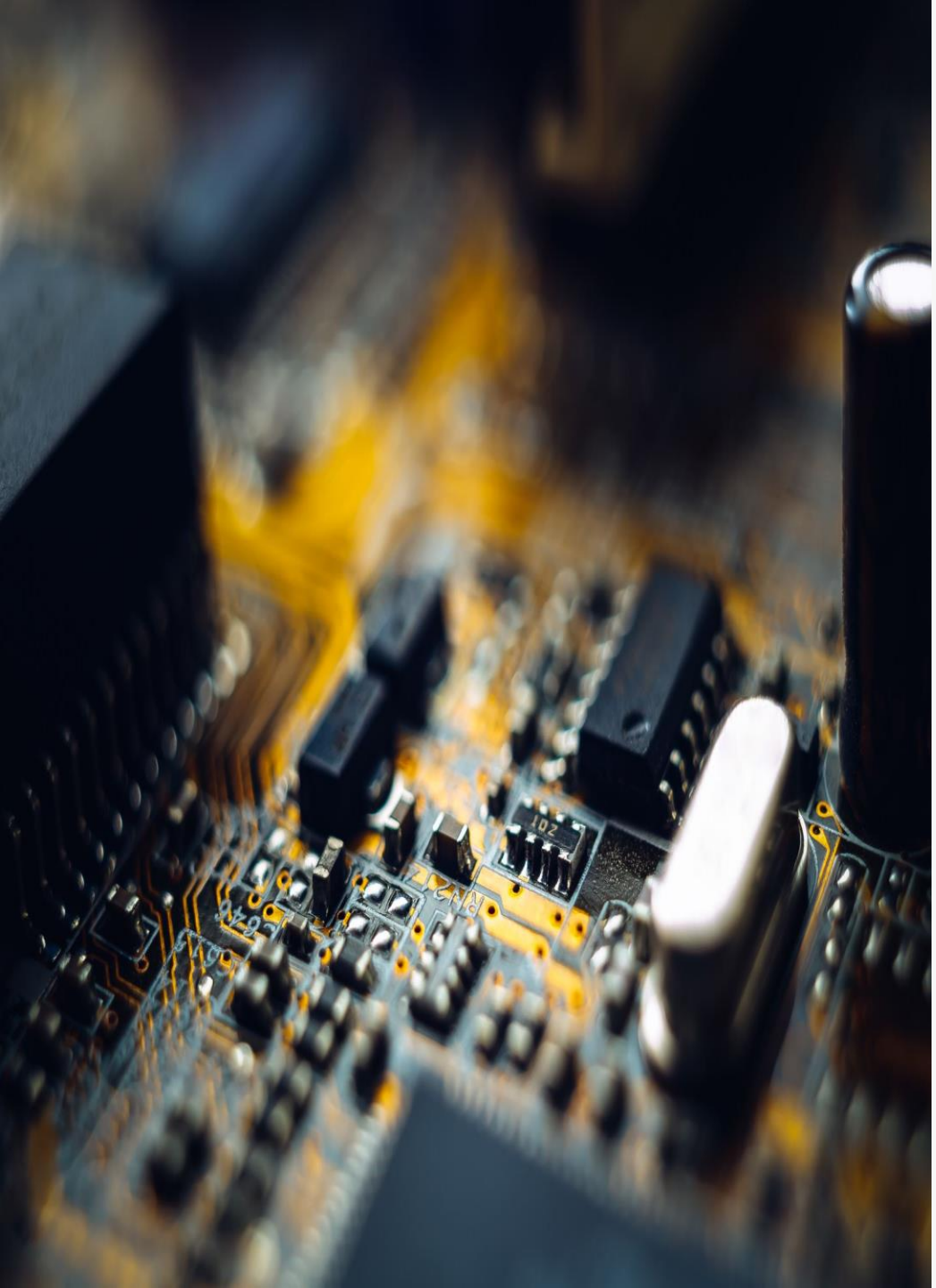
Performing Data collection: Python and Jupyter notebooks are leveraged along with key Python libraries (Pandas), APIs, web scraping, parsing of html, data pre-processing and cleaning, and exporting data in csv format for further use.

Performing data wrangling: Importing and working with python libraries, handling missing values, categorizing columns, calculating launch site and orbit distributions, determining mission outcomes, creating a success label, and general feature engineering.

Performing exploratory data analysis (EDA) using visualization and SQL: Using SQL queries and visualizations, including plotting charts. Feature engineering steps involved creating dummy variables and casting numeric columns.

Performing interactive visual analytics using Folium and Plotly Dash: Interactive visual analytics using Folium for geographical representation with markers, circles, and lines. Incorporating Plotly Dash for dynamic, web-based visualisations such as a Success Pie Chart and Success-Payload Scatter Chart.

Performing predictive analysis using classification models: Building, tuning, and evaluating machine learning classification models (logistic regression, SVM, Decision Tree, and KNN) through library imports, data preparation, and the use of grid search to optimize hyperparameters, ultimately selecting the best-performing model (KNN) based on multiple validation metrics.



Data Collection

Overview: The report's analysis uses python within a Jupyter notebook to collect and structure data, focusing on SpaceX Falcon 9 rocket launches to enable machine learning based predictive analytics on first-stage landing success. Key steps for collection included:

API Requests: Utilized SpaceX API for sourcing detailed launch information.

Data Filtering: Selected relevant data to support feature engineering, and filtered irrelevant values for analysis.

Helper Functions: Developed Python functions for targeted API calls, extracting relevant launch details for analysis.

Data Pre-processing: Executed functions to collect data, storing results in global variables (lists), and constructed a structured dictionary.

Dataframe Creation: Employed Pandas to create a dictionary from the lists, and converted these into a Dataframe (**launch_df**) for clarity and analysis.

Filtering Falcon 9 Launches: Exclusively focused on Falcon 9 launches, resulting in the creation of the **data_falcon9** dataframe.

Handling Missing Values: Addressed NaN values in **PayloadMass** by replacing them with the mean of existing data.

Data Export: Saved the final Falcon 9 dataset (**data_falcon9**) as a CSV file (**dataset_part_1.csv**) for use.

Data Collection
Flowchart of SpaceX REST API call process

The analysis for this report builds on the collection of data using the SpaceX REST API. A detailed flowchart of the SpaceX REST call process used for the collection process is provided below.

Flowchart of SpaceX REST API call process

- Import libraries and define auxiliary functions
- Make API call (using GET request) to request and parse SpaceX launch data
- Create a subset of the dataframe to only include key columns ('rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc')
- Create empty lists to store the results of auxiliary functions as global variables
- Call auxiliary functions (automatically writing outputs to the lists, as defined in the auxiliary functions)
- Use stored results of auxiliary functions (lists) to create a new dictionary
- Use that dictionary to create a new dataframe
- Data wrangling: Apply further cleansing and preprocessing to deal with irrelevant and missing data to finalise dataframe for use

NOTE: Please note that subtasks including validation of processes can be viewed in greater detail in the notebook provided using the GitHub URL.

Please use this link to the notebook for the REST API call process at the GitHub repository for this project: https://github.com/JJConrick/IBM-Data-Science-Capstone-Project-Final-Report/blob/15a71452ec3149fd6018c6bb3e26be8eee81298b/jupyter-labs-spacex-data-collection-api%20Lab%201%20-%20J%20Conrick.ipynb

Data Collection
Flowchart of web scraping process

The analysis for this report also builds on the collection of data using web scraping processes to source data from the SpaceX Wikipedia page. A detailed flowchart of the web scraping process used for the collection process is provided below.

Flowchart of web scraping process

- Import libraries and define auxiliary functions
- Request the Falcon9 Launch Wiki page from its URL using HTTP GET method
- Create a BeautifulSoup object from the HTML response
- Extract all column/variable names from the HTML table header
- Create a dictionary to populate data frame by parsing the launch HTML tables
- Define function to parse all launch tables into dictionary
- Create dataframe from dictionary
- Export final dataframe to csv for use

Please use this link to the notebook for the REST API call process at the GitHub repository for this project: https://github.com/JJConrick/IBM-Data-Science-Capstone-Project-Final-Report/blob/15a71452ec3149fd6018c6bb3e26be8eee81298b/jupyter-labs-webscraping-2%20-%20J%20Conrick.ipynb

Data Wrangling Process
Flowchart of data wrangling process

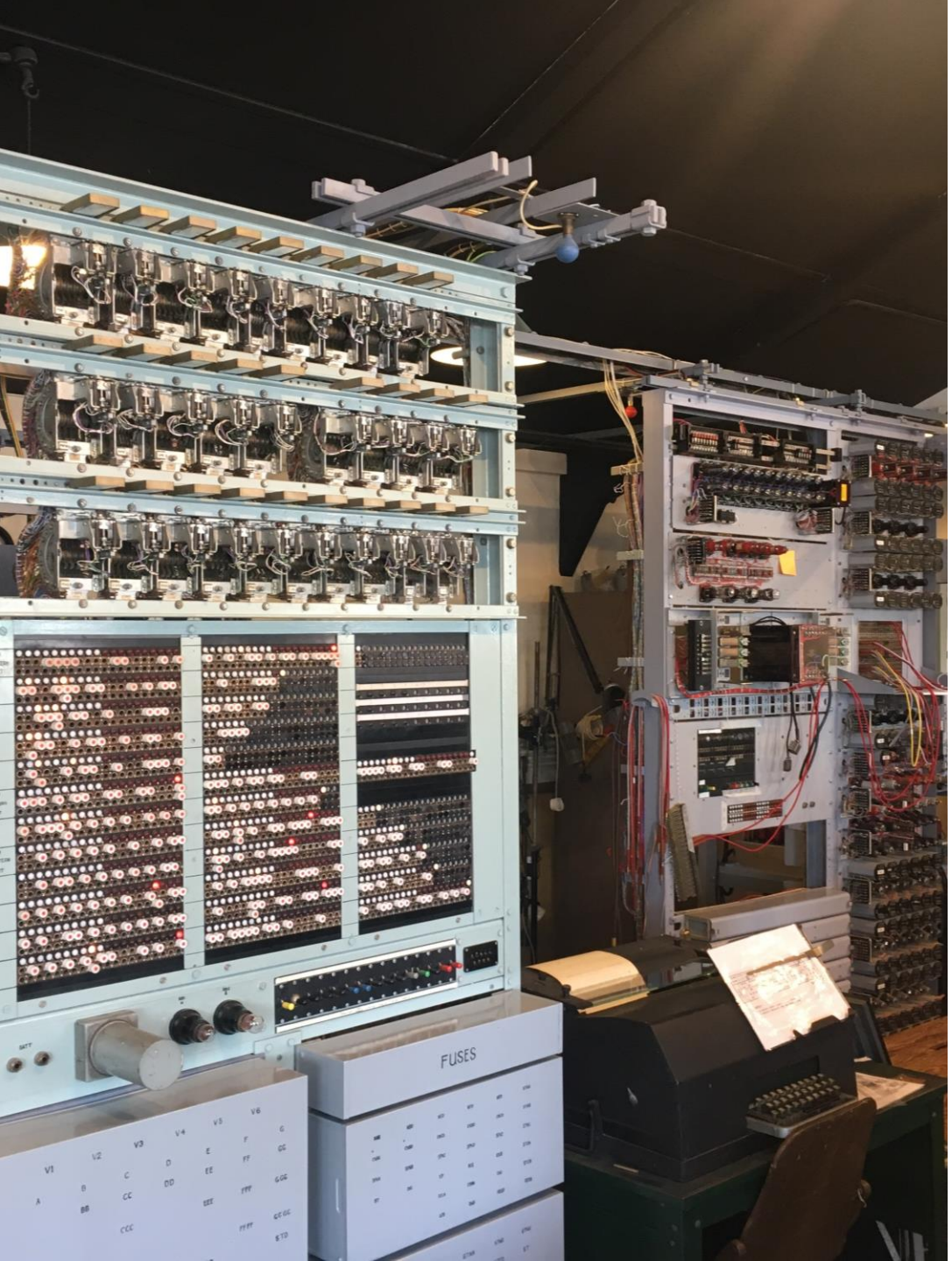
The analysis for this report relies on data wrangling processes being applied to make the data useful for analysis. A detailed flowchart of the data wrangling process used for the project is provided opposite.

Flowchart of data wrangling process

- Import libraries and define auxiliary functions
- Identify and calculate the percentage of the missing values in each attribute
- Identify which columns are numerical and categorical:
- Calculate the number of launches on each site
- Calculate the number and occurrence of each orbit
- Calculate the number and occurence of mission outcome of the orbits
- Create a set of outcomes where the second stage did not land successfully
- Create a landing outcome label from Outcome column
- Export dataframe to csv format for use

Please use this link to the notebook for the REST API call process at the GitHub repository for this project:

https://github.com/JJConrick/IBM-Data-Science-Capstone-Project-Final-Report/blob/c7c58e0da1d92346b684a2c2c3362fb399f21f68/jupyter-spacex-Data%20wrangling%20-%20J%20Conrick.ipynb



Exploratory Data Analysis with SQL

The SQL queries (using SQL Magic/SQLalchemy) used in the project include:

- Created table where dates were not null
- Displayed unique launch site names using SQL DISTINCT.
- Listed 5 records with launch sites starting with 'CCA' using SQL LIKE.
- Calculated total payload mass for NASA (CRS) launches using SQL SUM.
- Found the average payload mass for booster version F9 v1.1 using SQL AVG.
- Identified the date of the first successful landing on a ground pad using SQL MIN.
- Listed boosters with success in a drone ship and payload mass between 4000 and 6000 kg using SQL conditions.
- Counted total successful and failure mission outcomes using SQL COUNT and GROUP BY.
- Identified booster versions with the maximum payload mass using a subquery in SQL.
- Listed records for failure landing outcomes in a drone ship for 2015 using SQL conditions.
- Ranked landing outcomes counts between specified dates using SQL ORDER BY and WHERE.

These SQL queries addressed various aspects of the SpaceX dataset, including data filtering, aggregation, and extraction of specific information based on different criteria.

Please use this link to the notebook for the EDA with SQL process at the GitHub repository for this project: https://github.com/JJConrick/IBM-Data-Science-Capstone-Project-Final-Report/blob/3231ad640fca93cb9aa10e098b7f4fa8bbbe6f02/labjupyter-labs-eda-sql-coursera_sqllite%20-%20J%20Conrick.ipynb

Exploratory Data Analysis with Data Visualization

The charts plotted and the purpose for those charts as part of the EDA data visualization process are outlined below:

- **Flight Number vs. Payload Mass:** To observe any discernible relationship between the continuous launch attempts (Flight Number) and Payload Mass, aiming to identify patterns.
- **Flight Number vs. Launch Site:** To visualize patterns in the relationship between Flight Number and Launch Site, seeking insights into how the launch site might influence success.
- **Payload vs. Launch Site:** To explore how Launch Site correlates with Payload, aiming to understand if certain launch sites are better suited for specific payload masses.
- **Success Rate of Each Orbit Type:** To illustrate the variation in success rates across different orbit types using a bar chart, aiding in the identification of orbits with higher success rates.
- **Flight Number vs. Orbit Type:** To visualize the relationship between Flight Number and Orbit Type, aiming to identify trends or patterns that might influence success.
- **Payload vs. Orbit Type:** To examine how Payload relates to Orbit Type, helping to understand if certain orbits are more suitable for specific payload ranges.
- **Launch Success Yearly Trend:** To depict the average success rate trend over the years, providing insights into the overall performance and improvement over time.
- This also included steps for feature engineering, including the following:
- **Create Dummy Variables (One-Hot Encoding):** To convert categorical columns (Orbit, Launch Site, LandingPad, Serial) into numerical format, facilitating machine learning model compatibility.
- **Cast Numeric Columns to float64:** To ensure that all numeric columns in the dataset are of the **float64** data type, meeting the requirements for machine learning algorithms.

Please use this link https://github.com/JJConrick/IBM-Data-Science-Capstone-Project-Final-Report/blob/8b7e7caedc27f42c57dc72f469f58a9ce6b35e29/IBM-DS0321EN-SkillsNetwork_labs_module_2_Final%20IBM-DS0321EN-SkillsNetwork_labs_module_2_jupyter-labs-eda-dataviz.ipynb.jupyterlite%20-%20J%20Conrick.ipynb

Building Interactive Maps With Folium

For this project, various Folium map objects, including markers, circles, and lines, were strategically utilized to enhance the visualization and analysis of SpaceX launch data:

- **Markers:** Represent launch sites, aiding in visualizing their distribution and strategic placement patterns. Clusters of markers may suggest regions with higher launch activity.
- **Circles:** Highlight areas of interest, such as space centers, providing a visual reference for key points on the map. Analyzing circle size and placement offers insights into the significance of specific locations.
- **Marker Clusters:** Manage multiple markers with the same coordinates, enhancing map readability by grouping closely located markers. Reveals insights into the frequency and consistency of launches from specific locations.
- **Lines (Polyline):** Connect points of interest, such as launch sites to coastlines, visually representing distances and geographical relationships. Provides a clear understanding of spatial patterns and accessibility of launch sites.
- In summary, these Folium map objects contribute to an interactive and informative visualization of SpaceX launch data, pinpointing launch sites, highlighting areas of interest, managing overlapping markers, and illustrating geographical relationships within the dataset.

Please use this link to the notebook for the building of interactive maps with folium at the GitHub repository for this project: https://github.com/JJConrick/IBM-Data-Science-Capstone-Project-Final-Report/blob/f4f9d499c26384ee73b89ac467604489d7d4f236/lab_jupyter_launch_site_location.jupyterlite%20-%20J%20Conrick.ipynb

Build a Dashboard with Plotly Dash

As part of the Dash application code developed for this project, the following plots/graphs and interactions have been added to a web-based dashboard that updates dynamically in line with any changes to the underlying data:

Plots/Graphs:

Success Pie Chart: Displays a pie chart showing the distribution of success counts for launches. The chart dynamically updates based on the selected launch site from the dropdown menu. This chart provides an overview of launch success counts, allowing users to quickly assess success ratios for different launch sites.

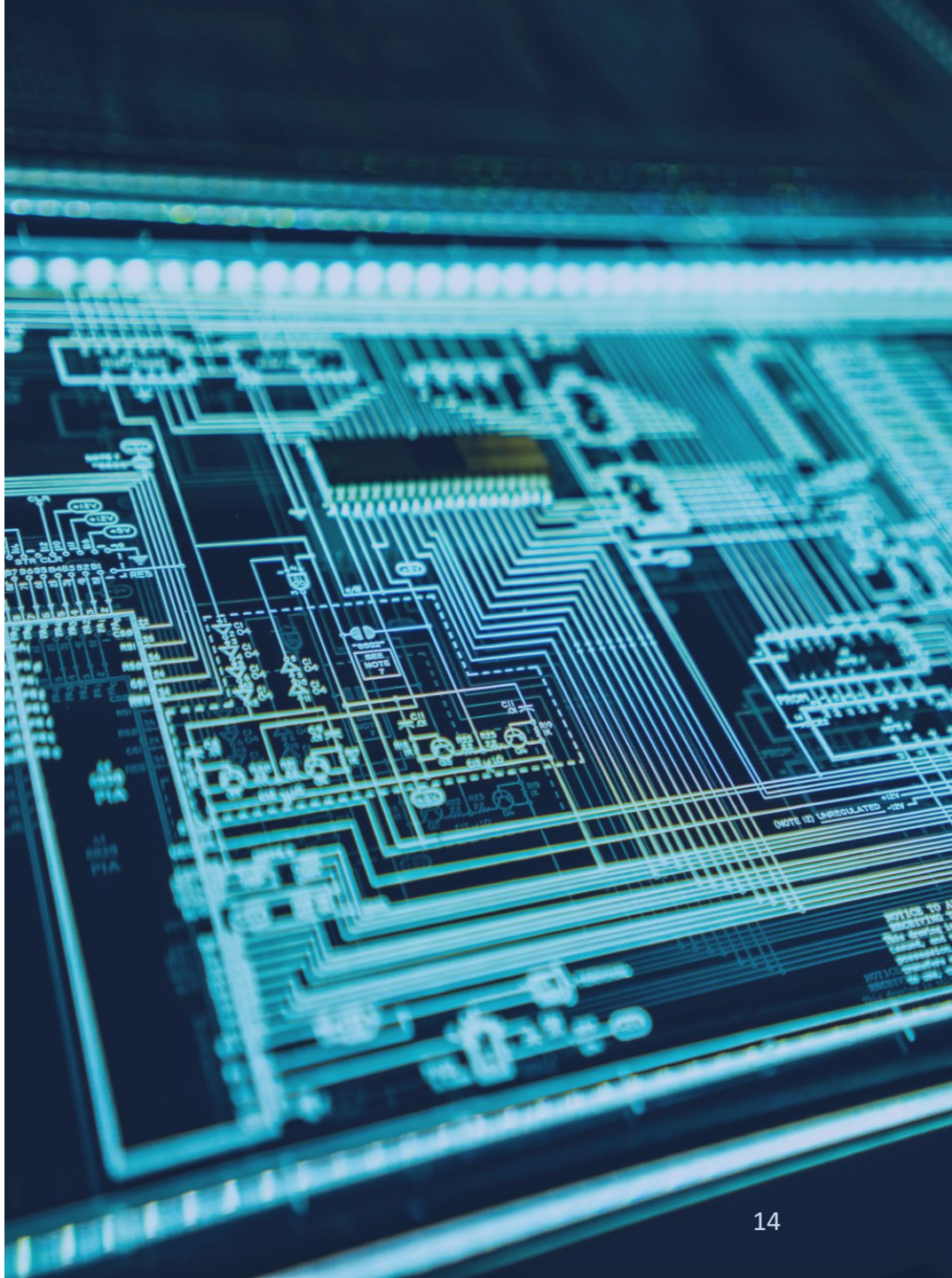
Success-Payload Scatter Chart: Shows a scatter chart depicting the correlation between payload mass and launch success. Updates based on the selected launch site from the dropdown and the payload range selected with the slider. This scatter chart helps explore any correlation between payload mass and launch success. It provides insights into whether certain payload ranges are more prone to success or failure and how this relationship varies across launch sites.

Interactions:

Dropdown for Launch Site Selection: Allows users to select a specific launch site or view data for all sites. Users can focus on a particular launch site or examine the overall success distribution for all sites. It provides flexibility in data exploration.

Slider for Payload Range Selection: Enables users to filter launches based on payload mass. Users can narrow down the data to specific payload ranges of interest, helping identify patterns in launch success based on payload mass.

Please use this link to the notebook for the building a dashboard with Plotly Dash process at the GitHub repository for this project: https://github.com/JJConrick/IBM-Data-Science-Capstone-Project-Final-Report/blob/47e577dd90bb274b57b27df341e7e4d79defaddd/Final%20spacex_dash_app%20notebook%20-%20J%20Conrick.py



Building a Dashboard with Plotly Dash continued

Overall Explanation:

- **Insightful Visualization:** The pie chart and scatter chart visually represent key aspects of SpaceX launch data, making it easier for users to grasp trends and relationships.
- **Interactivity for Exploration:** The dropdown menu and slider allow users to interactively explore data based on specific criteria. This interactivity enhances the user experience and facilitates more focused analysis.
- **Comparative Analysis:** The charts facilitate a comparative analysis of success counts among different launch sites and the relationship between payload mass and launch success.
- **Decision Support:** Users can use the dashboard to make informed decisions or hypotheses about factors influencing launch success, such as the influence of payload mass or the performance of specific launch sites.

In summary, the plots and interactions added to the dashboard provide a comprehensive and interactive exploration of SpaceX launch data, empowering users to analyze success patterns and relationships with payload mass across different launch sites.

Predictive Analysis (Classification)

The report's predictive analysis involved building, optimizing, and tuning machine learning models—logistic regression, SVM, Decision Tree, and KNN—using pandas and scikit-learn. Hyperparameter optimization employed the GridSearch function. The best classification model (KNN) was chosen based on a coded validation process, comparing overall scores on multiple metrics.

This approach included rigorous data preparation, standardization, and systematic model evaluation.

The GridSearch function played a crucial role in hyperparameter tuning, with KNN identified as the most effective model through a comprehensive comparison of performance metrics.

Flowchart of Predictive Analysis Classification Process

Key steps for the predictive analysis process for this project included:

- **Imported Libraries and Defined Auxiliary Functions:** Prepared the groundwork by importing necessary libraries and defining auxiliary functions, particularly for the confusion matrix.
- **Data Preparation:** Loaded the dataframe and assigned columns to the X variable. Created a NumPy array from the 'Class' column, assigning it to the Y/Target variable.
- **Train-Test Split:** Applied the train-test split to ensure robust model evaluation.
- **Pre-processing:** Applied standardization (using the standard scalar function) to the X object. Validated outcomes of the standardization process.
- **Model Development, Hyper-parameter Optimization, and Tuning:** Developed, fit, trained, and tuned logistic regression, SVM, Decision Tree, and K Nearest Neighbour models using the gridsearch function. Identified optimized hyperparameters through gridsearch.
- **Model Evaluation and Selection:** Validated logistic regression against accuracy score and confusion matrix. Conducted a series of tests to identify the optimal model, focusing on K-Nearest Neighbors.

Please use this link : https://github.com/JJConrick/IBM-Data-Science-Capstone-Project-Final-Report/blob/a4cb3683d53acb854e06c028af74f4ebc69ae116/SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite%20-%20J%20Conrick.ipynb

Section Four

Insights drawn from Exploratory Data Analysis

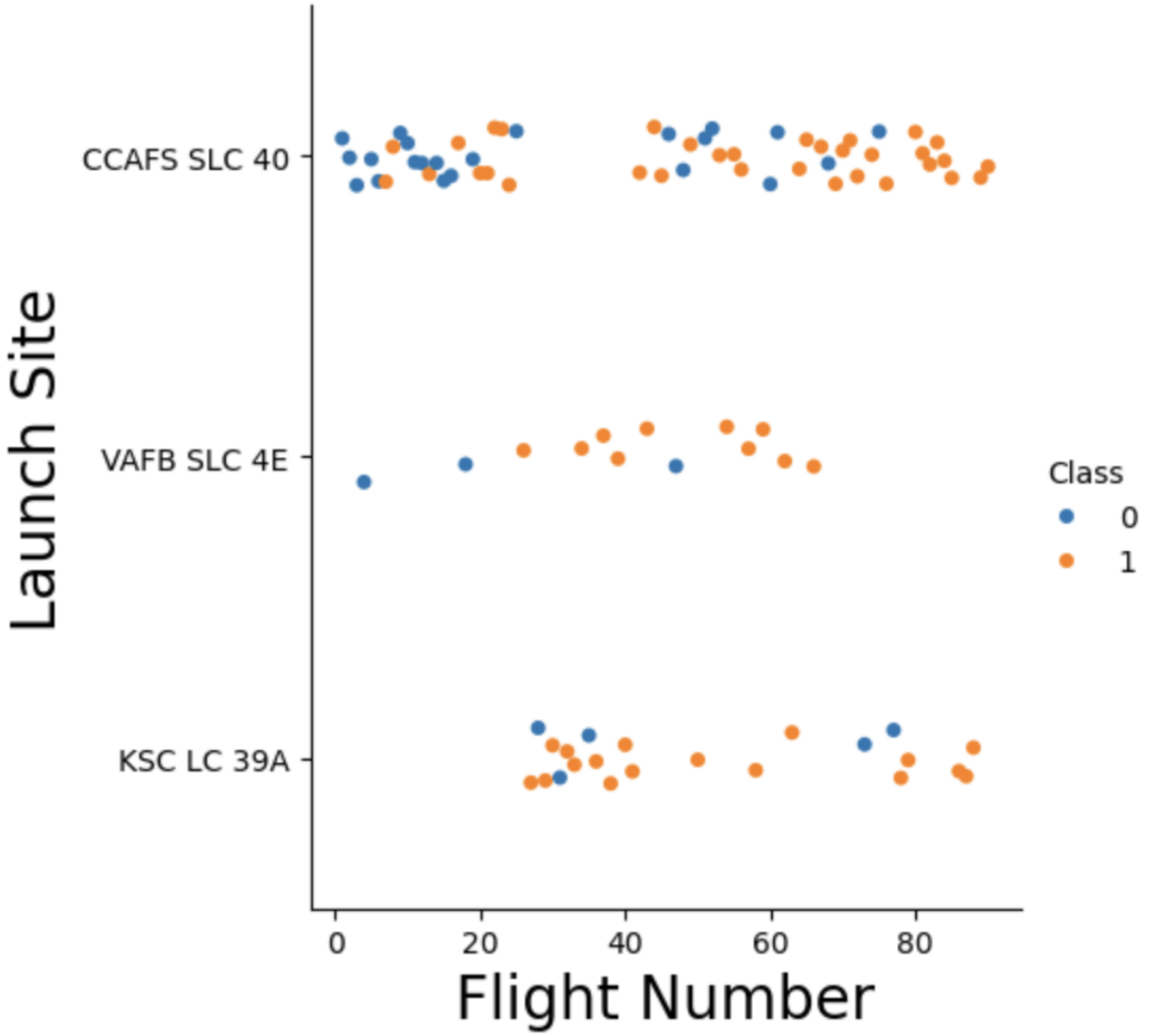
Flight Number vs. Launch Site

This shows the overall flight number for SpaceX rocket launches plotted against three key launch sites.

We can see from the plot that launch success has increased over time, and that earlier launches with a higher failure rate were concentrated at site CCAFS SLC 40.

CCAFS SLC 40 continues to host a large proportion of SpaceX's overall launches, but with a higher success rate than in the first quartile.

A smaller proportion of overall launches have been conducted at VAFB SLC 4E and KSC LC 39A with higher success rates overall than were posted in the first quartile of launches at CCAFS SLC 40.



Payload vs. Launch Site

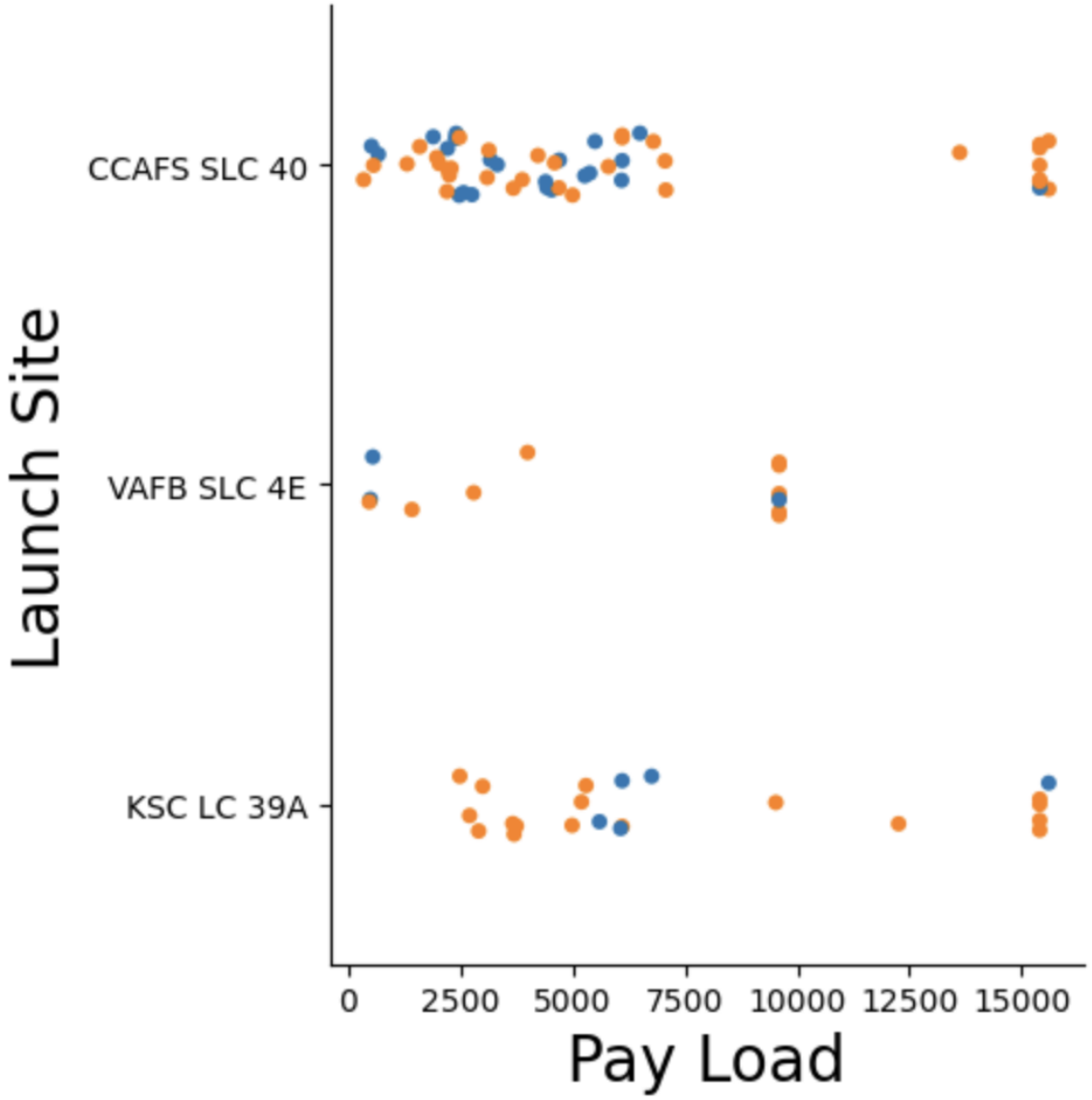
This shows the payload for SpaceX rocket launches plotted against each launch site.

We can see from the plot that payloads and success rates for higher payloads have generally increased over time.

Again, we can see early experimentation at CCAFS SLC 40 leading to improved success rates and diversification into different payload products.

The uneven distribution of payloads across sites clearly shows how SpaceX has targeted different products based on each location's unique geospatial aspects.

VAFB SLC 4E is shown to support less and lower weight launches overall, and CCAFS SLC 40 and KSC LC 39A are shows to share a range of payload weight classes.



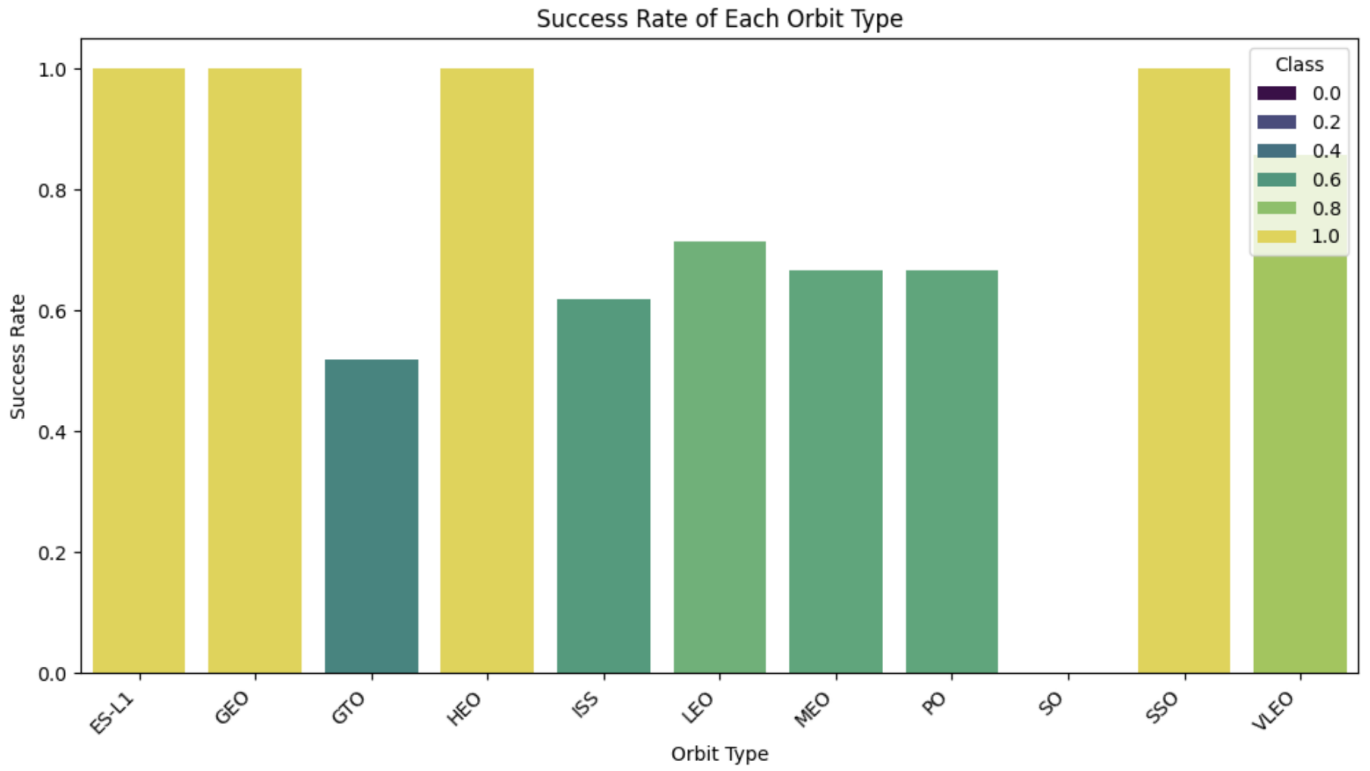
Success Rate vs. Orbit Type

This bar graph shows the overall standardised success rate for SpaceX rocket launches plotted against Orbit types.

We can see from the chart that ESL-1, GEO, HEO and SSO have the highest success rates, followed by VLEO and the group of LEO, MEO, PO, ISS and GTO, with SO the only orbit type with no success.

While this data has been standardised, we note that overall success rates here should not necessarily be used in isolation to draw conclusions on likely product success, as some scores only represent one successful landing (i.e. ESL-1, HEO), and were also achieved later in the overall life of the SpaceX program following extensive product testing. Other orbit types have lower overall scores as they are higher volume products for SpaceX and where also tested heavily in early experimentation phases leading to overall lower scores (GTO for example). There may also be other reasons for score variation not present in the data and requiring further analysis.

Using appropriately selected and balanced machine learning models will allow support users in drawing insights from key relevant information in the dataset to predict launch outcomes.

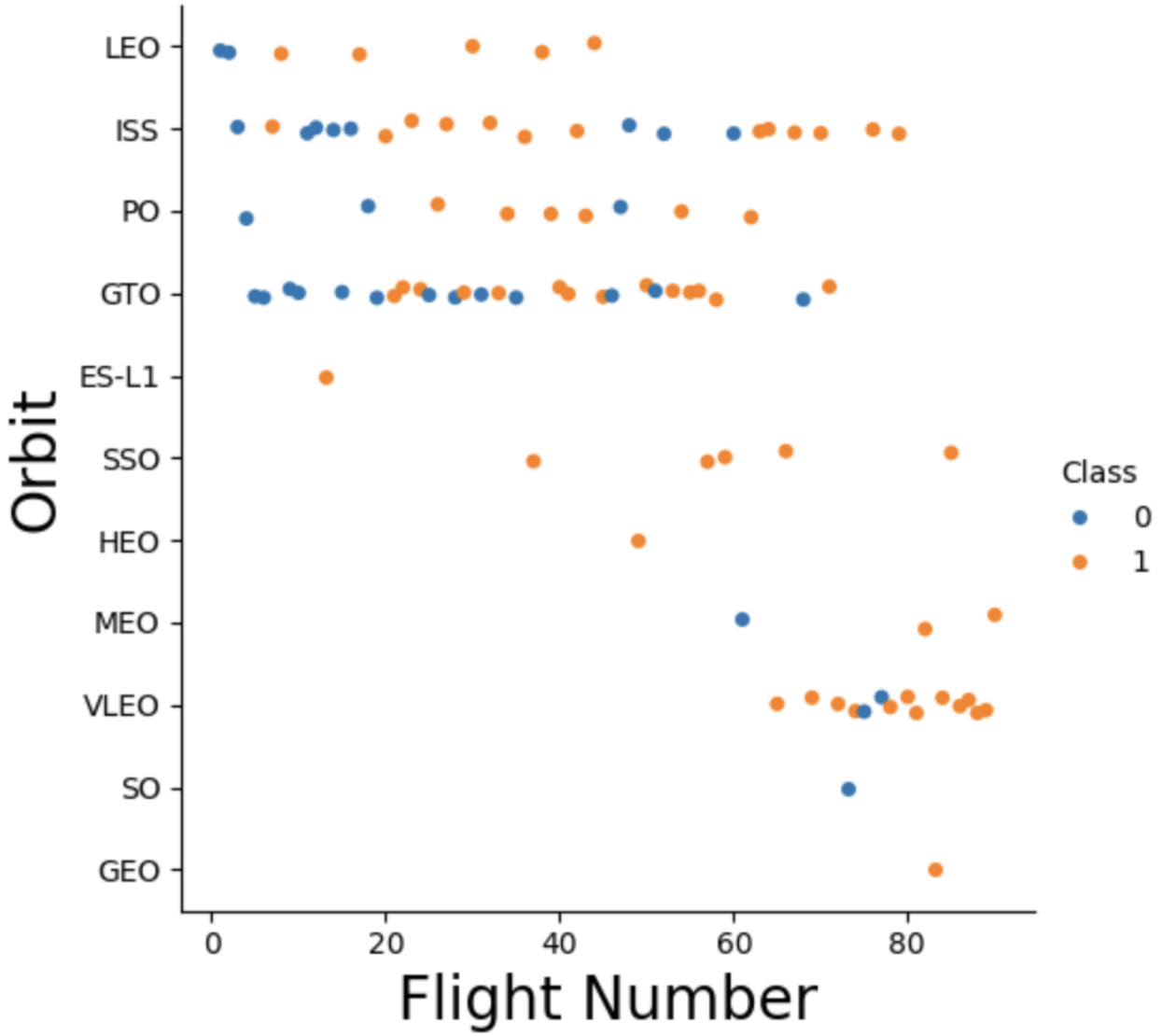


Flight Number vs. Orbit Type

This shows the orbit type for launches against the overall flight number.

We can see that overall launch success has increased over time, as well as variation in orbit type over time.

It seems that some new orbit types are released with higher success rates following initial testing in other earlier orbit types.



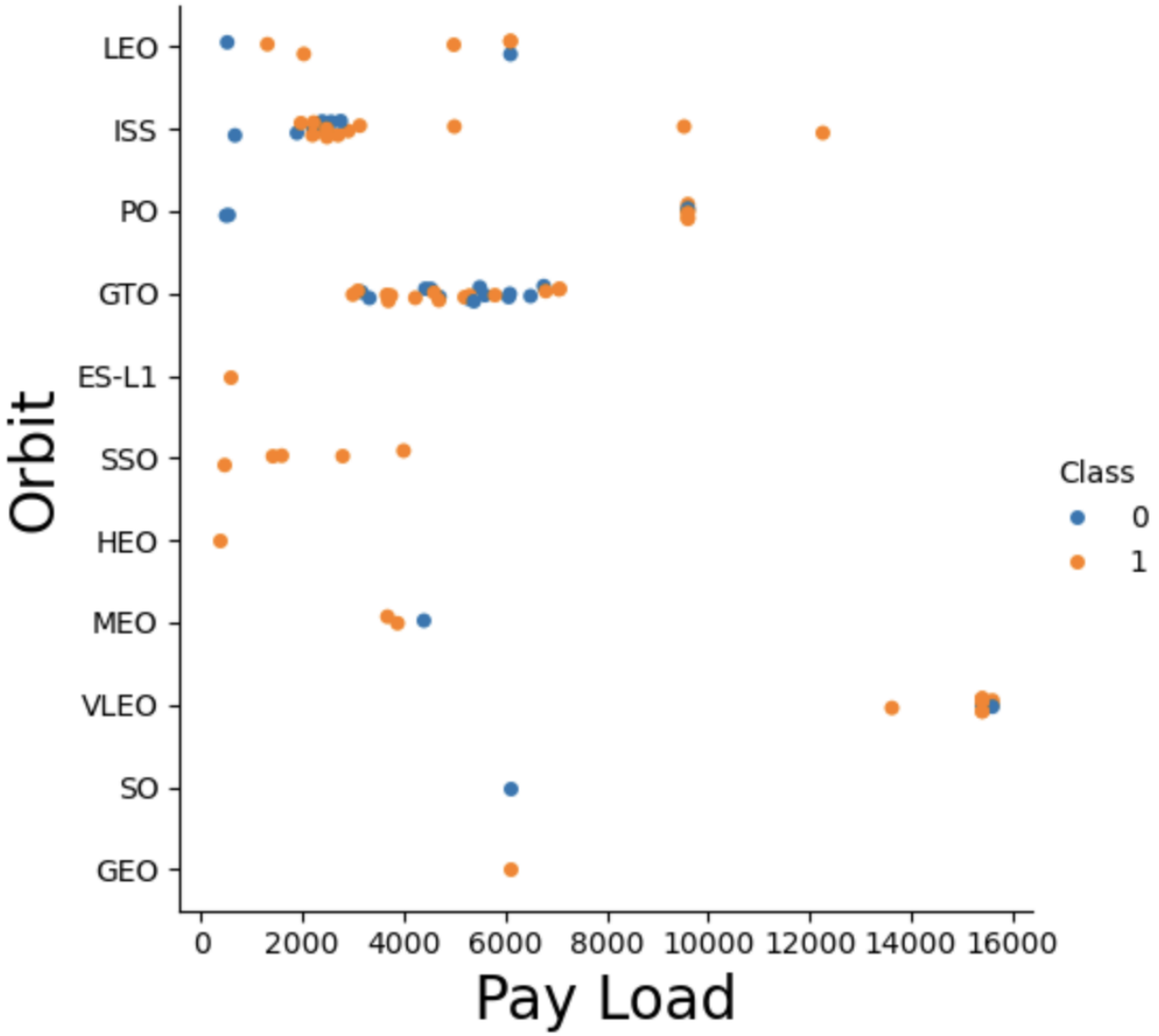
Payload vs. Orbit Type

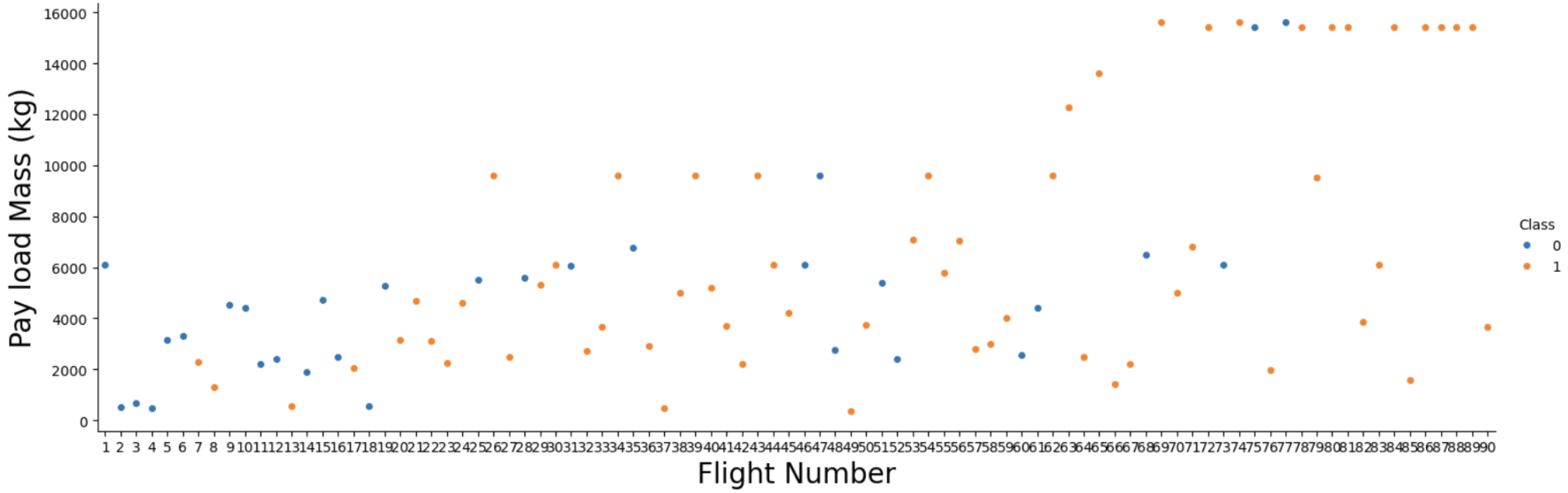
This shows the payload vs the orbit type and should be considered in conjunction with the payload vs flight number plot provided on the next slide.

Considered together, we can see that payload weight and orbit type increased progressively over time.

Certain orbit types have higher launch volumes overall, and higher early testing in the life of the program (GTO, ISS, etc).

Each orbit type has a distict payload range (i.e. lower for SSO).





Payload vs. Flight Number

This plot shows payload vs flight number. As mentioned above, we can see an overall gradual increase in payloads and success rates over time.

We can also observe patterns in consistent payloads delivered over time which indicates clear product lines for SpaceX, and their success rates.

This will be useful for targeting product development and pricing strategies if bidding against SpaceX.

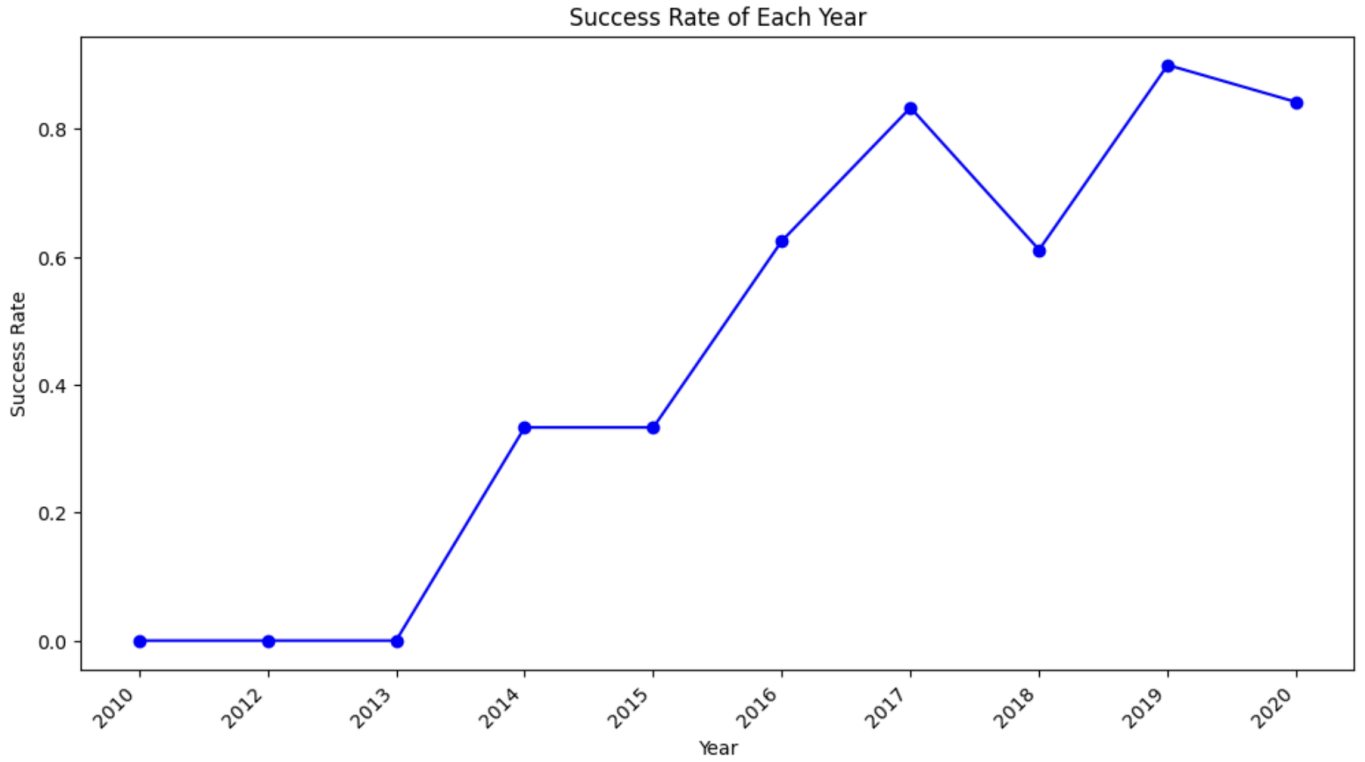
Launch Success Yearly Trend

This plot line graph shows success rate vs year.

It shows a clear increase in success rate over time.

We can observe higher failure rates in earlier years, as well as periods of failure increase at certain inflection points (i.e. 2018).

Further analysis is possible to show reasons for failure rates throughout, testing whether this was caused because of testing new products, payloads, orbit types, etc.



All Launch Site Names

This image shows the SQL (SQLLite) query to determine the unique launch sites.

As can be seen, there is variation from earlier analysis, with the introduction of the closely located CCAFS LC-40 site.

We use the select, distinct, and from clauses to filter data from the SPACEXTABLE.

Display the names of the unique launch sites in the space mission using sql magic

In [16]:
```
%%sql

SELECT DISTINCT "Launch_Site"
FROM SPACEXTABLE;
```

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

Out[16]:

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

Launch Site Names Begin with 'CCA'

This image shows the SQL query used to determine launch sites with names that begin with 'CCA'.

We make use of the select, from, where, like and limit clauses to filter data from SPACEXTABLE.

Task 2

Display 5 records where launch sites begin with the string 'CCA'

In [9]:

```
%%sql

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

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

Out[9]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_
6/4/2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0
12/8/2010	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0
22/05/2012	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525
10/8/2012	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500
3/1/2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677

Total Payload Mass

This image shows the SQL query used to determine the total payload carried by boosters from NASA.

We make use of the select, sum, as, from, where, and like clauses to filter data from SPACEXTABLE.

The total is 48213.

Task 3

Display the total payload mass carried by boosters launched by NASA (CRS)

In [10]:
```
%%sql

SELECT SUM("Payload_Mass__kg_") AS "Total_Payload_Mass_NASA_CRS"
FROM SPACEXTABLE
WHERE "Customer" LIKE 'NASA (CRS)%';
```

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

Out[10]:

Total_Payload_Mass_NASA_CRS
48213

Average Payload Mass by F9 v1.1

This image shows the SQL query used to determine the total payload carried by booster version F9 v1.1.

We make use of the select, sum, as, from, where, and like clauses to filter data from SPACEXTABLE.

The total is 2928.4.

Task 4

Display average payload mass carried by booster version F9 v1.1

In [11]:

```
%%sql

SELECT AVG("Payload_Mass__kg_") AS "Average_Payload_Mass_F9_v1.1"
FROM SPACEXTABLE
WHERE "Booster_Version" = 'F9 v1.1';
```

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

Out[11]:

Average_Payload_Mass_F9_v1.1
2928.4

First Successful Ground Landing Date

This image shows the SQL query used to determine the first successful landing outcome on ground pad.

We make use of the select, min, as, from, and where clauses to filter data from SPACEXTABLE.

The date is 1/8/2018.

Task 5

List the date when the first succesful landing outcome in ground pad was acheived.

Hint:Use min function

In [12]:
```
%%sql

SELECT MIN("Date") AS "Date_of_First_Successful_Landing_on_Ground_Pad"
FROM SPACEXTABLE
WHERE "Landing_Outcome" = 'Success (ground pad)';
```

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

Out[12]:

Date_of_First_Successful_Landing_on_Ground_Pad
1/8/2018

Successful Drone Ship Landing With Payload Between 4000 and 6000

This image shows the SQL query used to determine names of the boosters which have successfully landed on a drone ship and have payload mass greater than 4000 but less than 6000.

Task 6

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

In [13]:

```
%%sql

SELECT "Booster_Version", "Landing_Outcome", "Payload_Mass__kg_"
FROM SPACEXTABLE
WHERE "Landing_Outcome" = 'Success (drone ship)'
  AND "Payload_Mass__kg_" > 4000
  AND "Payload_Mass__kg_" < 6000;
```

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

Out[13]:

Booster_Version	Landing_Outcome	PAYLOAD_MASS__KG_
F9 FT B1022	Success (drone ship)	4696
F9 FT B1026	Success (drone ship)	4600
F9 FT B1021.2	Success (drone ship)	5300
F9 FT B1031.2	Success (drone ship)	5200

Total Number of Successful and Failure Mission Outcomes

This image shows the SQL query used to determine the total number of successful and failure mission outcomes.

Total successes are 100 (noting payload status for 1 launch unclear) and that the low failure number of 1 is likely due to this dataset being a subset for purposes of this project.

Task 7

List the total number of successful and failure mission outcomes

In [14]:

```
%%sql

SELECT "Mission_Outcome", COUNT(*) AS "Outcome_Count"
FROM SPACEXTABLE
GROUP BY "Mission_Outcome";
```

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

Out[14]:

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

Boosters Carried Maximum Payload

This image shows the SQL query used to determine the names of the booster which have carried the maximum payload mass.

Task 8

List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

In [15]:

```
%%sql

SELECT "Booster_Version"
FROM SPACEXTABLE
WHERE "Payload_Mass__kg_" = (SELECT MAX("Payload_Mass__kg_") FROM SPACEXTABLE);
```

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

Out[15]:

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

This image shows the SQL query used to determine the failed landing outcomes in drone ship, their booster versions, and launch site names for in year 2015.

Task 9

List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.

Note: SQLLite does not support monthnames. So you need to use substr(Date, 6,2) as month to get the months and substr(Date,0,5)='2015' for year.

In [17]:

```
%%sql

SELECT
    substr('JanFebMarAprMayJunJulAugSepOctNovDec', 1 + 3 * CAST(substr(Date, 4, 2) AS INT), 3) AS "Month",
    "Landing_Outcome",
    "Booster_Version",
    "Launch_Site"
FROM SPACEXTABLE
WHERE (
        substr(Date, 7, 4) = '2015'
        OR substr(Date, 6, 4) = '2015'
    )
    AND "Landing_Outcome" = 'Failure (drone ship)'
ORDER BY CAST(substr(Date, 4, 2) AS INT);
```

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

Out[17]:

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

2015 Launch Records Rank Landing Outcomes Between 2010-06-04 and 2017-03-20

This image shows the SQL query used to rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order

Task 10

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

In [21]:

```
%%sql

SELECT
    "Landing_Outcome",
    COUNT(*) AS "Outcome_Count"
FROM SPACEXTABLE
WHERE substr(Date, 7, 4) || '-' || substr(Date, 4, 2) || '-' || substr(Date, 1, 2) BETWEEN '2010-06-04' AND '2017
GROUP BY "Landing_Outcome"
ORDER BY "Outcome_Count" DESC;
```

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

Out[21]:

Landing_Outcome	Outcome_Count
Success (ground pad)	3
Success (drone ship)	3
No attempt	3
Failure (drone ship)	3
Uncontrolled (ocean)	2
Controlled (ocean)	2
Precluded (drone ship)	1

Section Three

Launch Sites Proximities Analysis

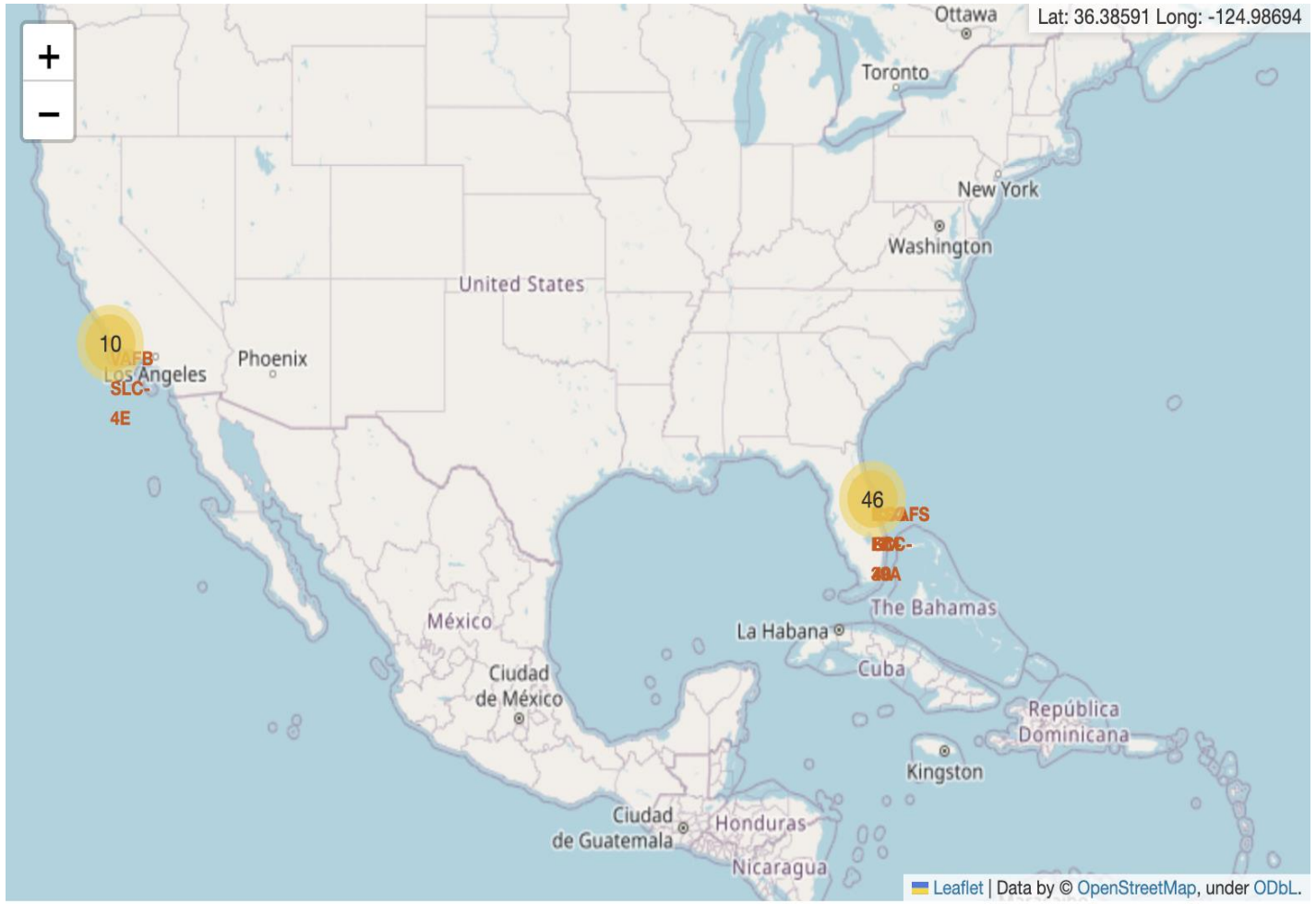
Folium Global Map – All Site Location Markers

This image shows the generated Folium global map screenshot including all launch sites' location markers.

Important elements and findings include close proximity for three sites on the East Coast, and a separate location on the West Coast for the other site.

We can also see that a majority of launches are made on the East Coast. Delving into orbit types and payloads, we know that only a limited range of launches are delivered on the West Coast (PO and SSO, with lighter payload).

We can see that all launch sites are relatively close to the equator, which provides advantages for certain types of launches. The closer a launch site is to the equator, the faster the rotational speed of the Earth at that location. This additional rotational speed can provide a boost to rockets during launch.



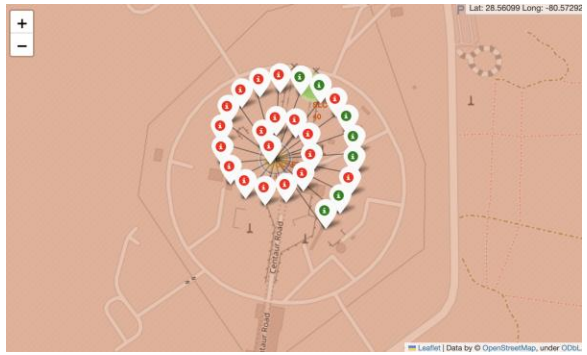
Folium Launch Sites Colour-Labeled Launch Outcomes

This image shows the Folium maps for each launch site with colour-labeled launch outcomes (red for failure, green for success.)

We can see that there is a spiral pattern to locations at a launch site, although we cannot tell why from the data. It may just be cheaper to host a new launch site, or it may affect the launch success. This would have to be analysed.

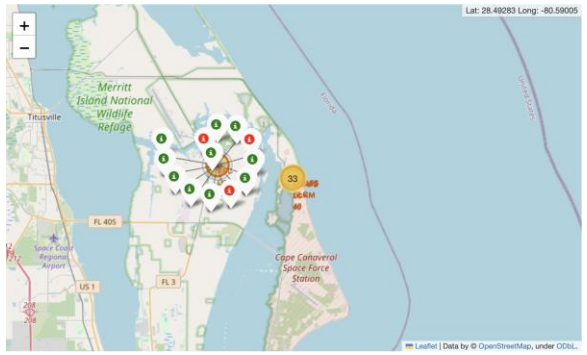
We can see clearly that there is a higher success rate at KSC LC 39A.

CCAFS SCL 40

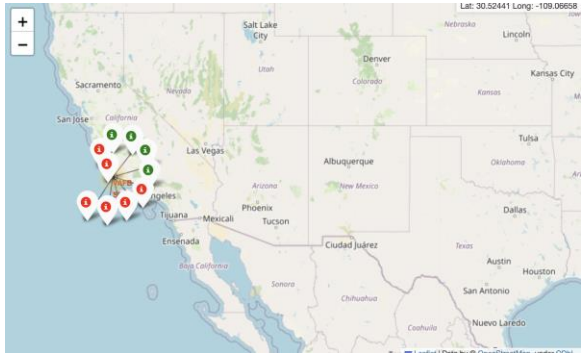


CCAFS SCL 40

KSC LC 39A



VAFB SLC 4E

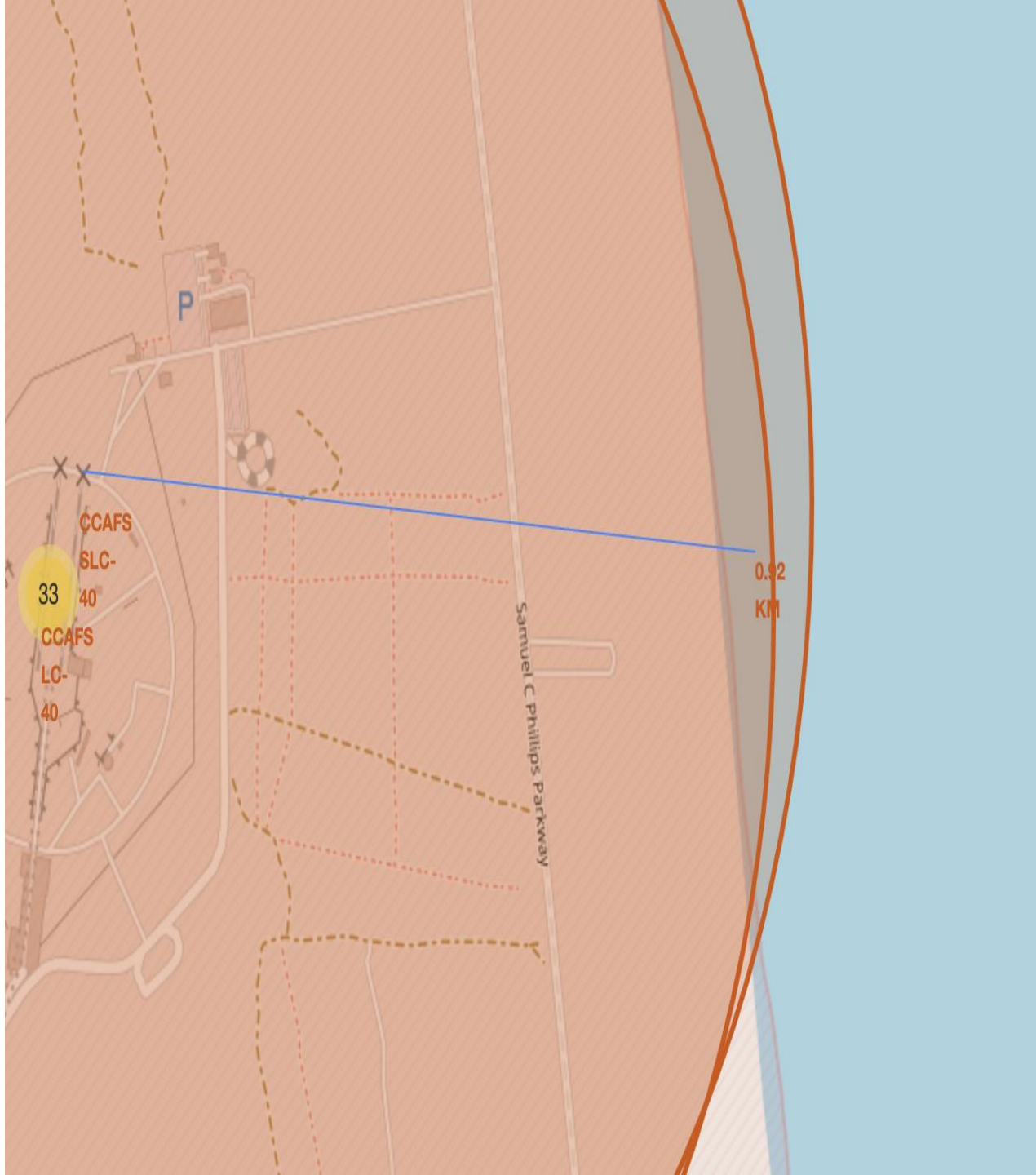


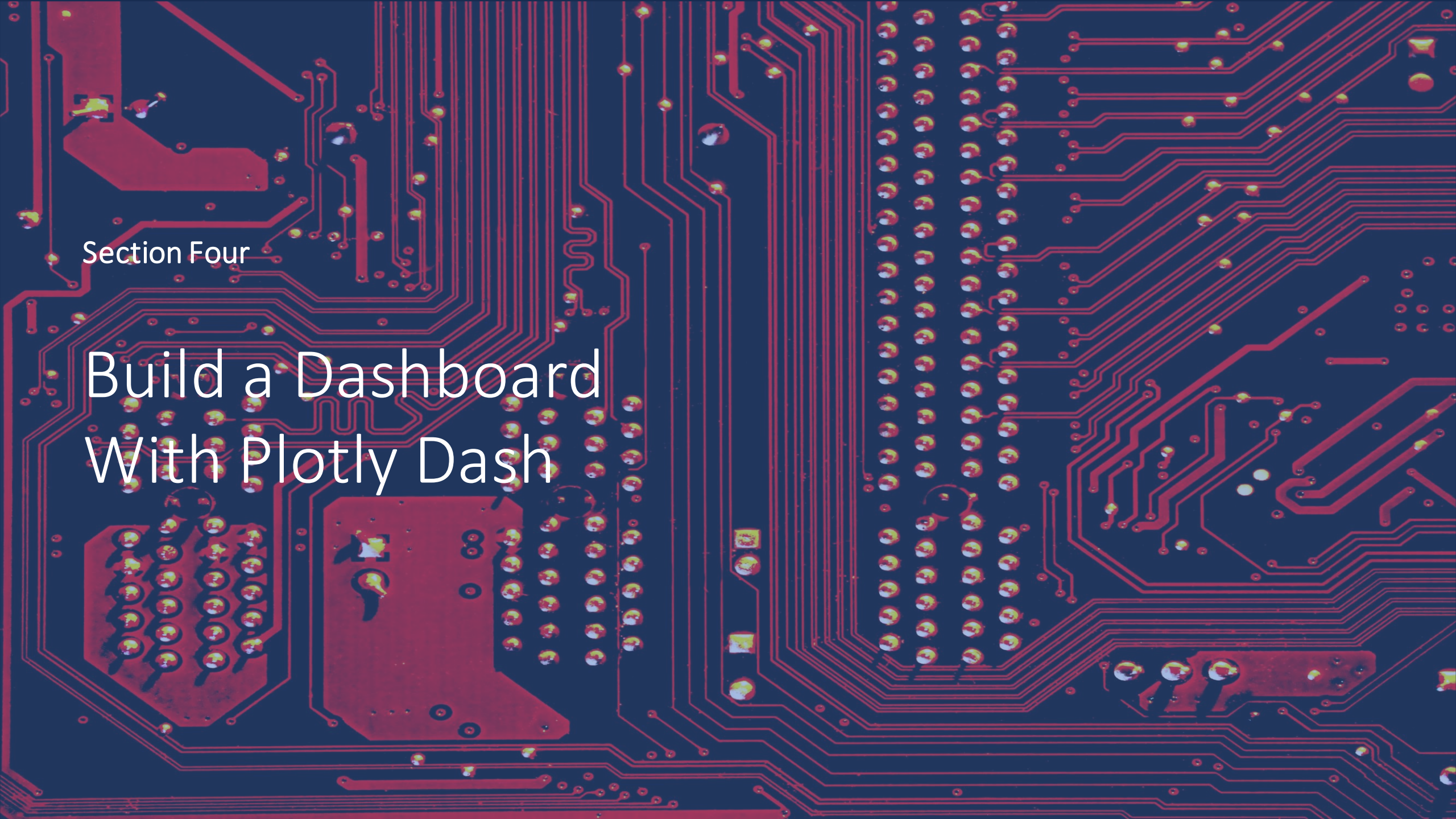
Folium Launch Sites Colour-Labeled Launch Outcomes

This image shows the generated Folium map and the proximity of the CCAFS SLC 40 site to the coastline, a distance of 0.92km.

Proximity to the ocean allows SpaceX a range of benefits, including:

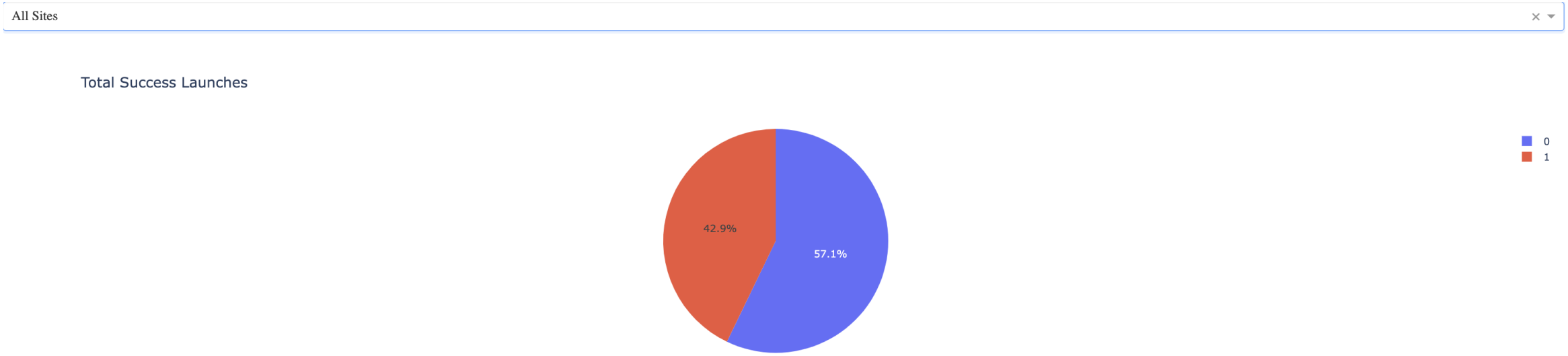
- Safety to land rockets in the ocean.
- Flexibility to vary flight in line with optimum trajectories without increasing safety risks.
- Ability to recover and potentially reuse rockets that didn't land successfully, but landed in the ocean.
- Lower transportation costs for rocket parts delivered by sea.
- Minimising pollution impacts of launches on general population.
- Meeting international safety regulations.





Section Four

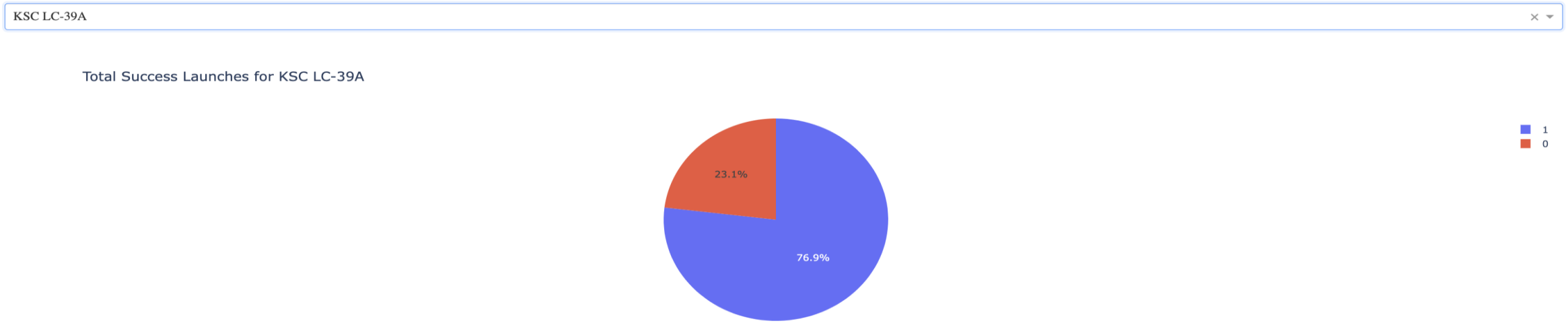
Build a Dashboard With Plotly Dash



Launch Success for All Sites

This image shows the percentage of the sum of overall success counts for all sites in a pie chart.

We can see that overall 42.9% of launches have succeeded, and 57.1% failed.



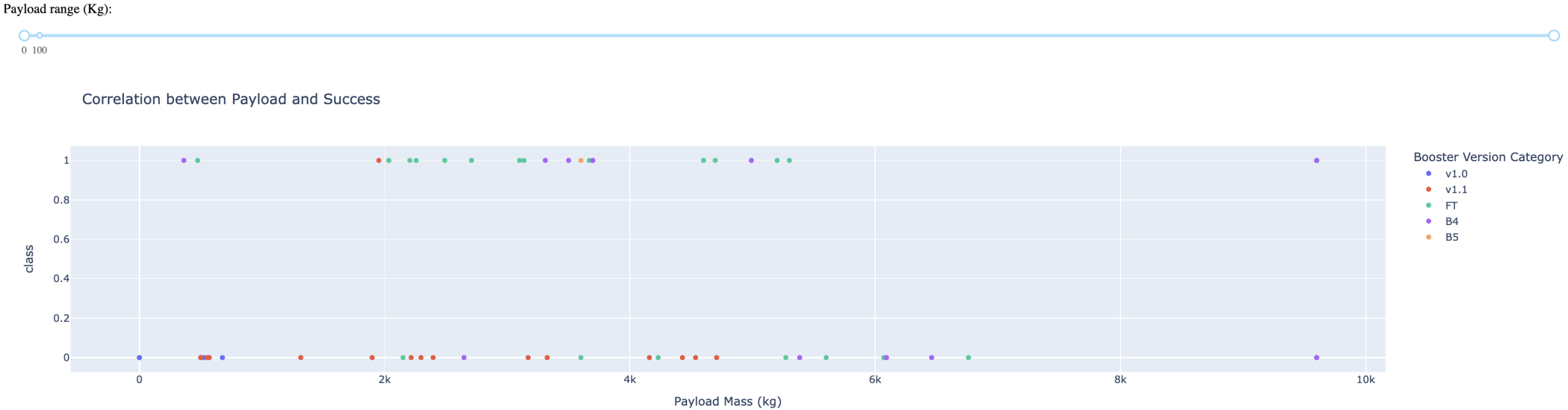
Site With Highest Launch Success Rate

This image shows the pie chart for the launch site with the highest launch success ratio, KSC LC-39A.

We can see that overall 76.9% of launches have succeeded, and only 23.1% failed.

We note this success is also likely due to successfully established products being launched at this site, which were tested thoroughly at other launch sites.

We also note that that this site is further inland – it may be that SpaceX moved safer launches to this site for proximity to urban areas for logistics and staff living arrangements, among other things.



Payload vs Launch Outcome for All Sites

This image shows the payload vs launch outcome ('Class') for all launch sites.

We can see that the FT Booster Version has the highest success rate.

We can see that the FT and B4 Booster Versions are the most numerous and are the only booster types to support heavier payloads (above 4k successfully).

We can see that the majority of payloads are between 2k and 6k.

Some booster versions clearly support lower payloads, likely for specific orbit types at specific sites.

Section Five

Predictive Analysis (Classification)

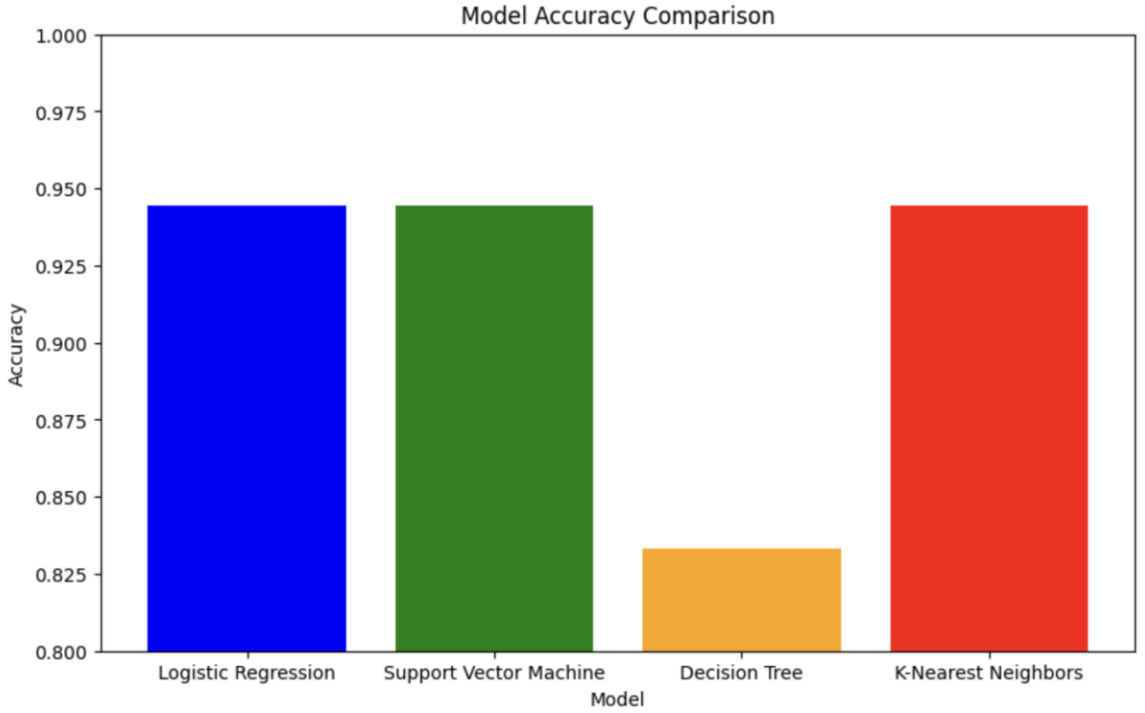
Classification Accuracy

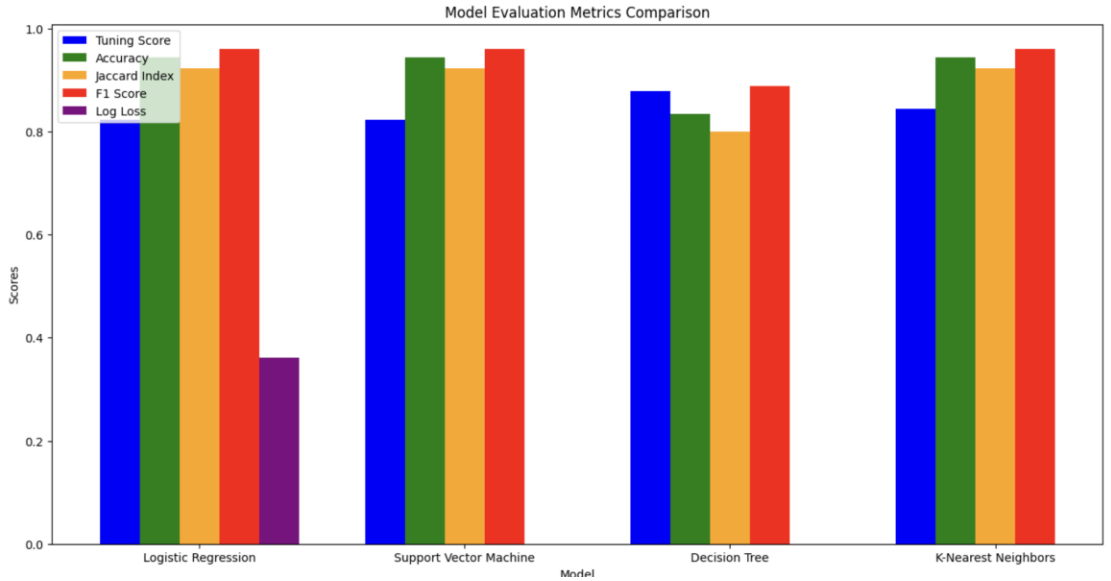
This image shows bar charts and tables depicting the accuracy of the machine learning classification models developed to predict the likelihood that a SpaceX rocket launch will be successful.

With Machine Learning key steps include training, evaluating and tuning to get the best models we can to predict in this case classification of the likelihood of an event.

To select K-Nearest-Neighbor as the best model, I have used additional validation metrics of tuning score, Jaccard Index, F1 Score and Log Loss.

While the best scoring model overall is K-Nearest Neighbor, I note that we can likely use an ensemble approach that uses multiple of these models to predict our answers. For example, the top models here might also be used together with gradient boosting or random forest algorithms to boost accuracy scores.





Metrics for Each Model:

	Model	Tuning Score	Accuracy	Jaccard Index	F1 Score	Log Loss
0	Logistic Regression	0.822222	0.944444	0.923077	0.960000	0.360438
1	Support Vector Machine	0.822222	0.944444	0.923077	0.960000	NaN
2	Decision Tree	0.877778	0.833333	0.800000	0.888889	NaN
3	K-Nearest Neighbors	0.844444	0.944444	0.923077	0.960000	NaN

```
The best-performing model (combined score) is: K-Nearest Neighbors
Hyperparameter Tuning Score: 0.8444444444444444
Accuracy on Test Data: 0.9444444444444444
```

Confusion Matrix

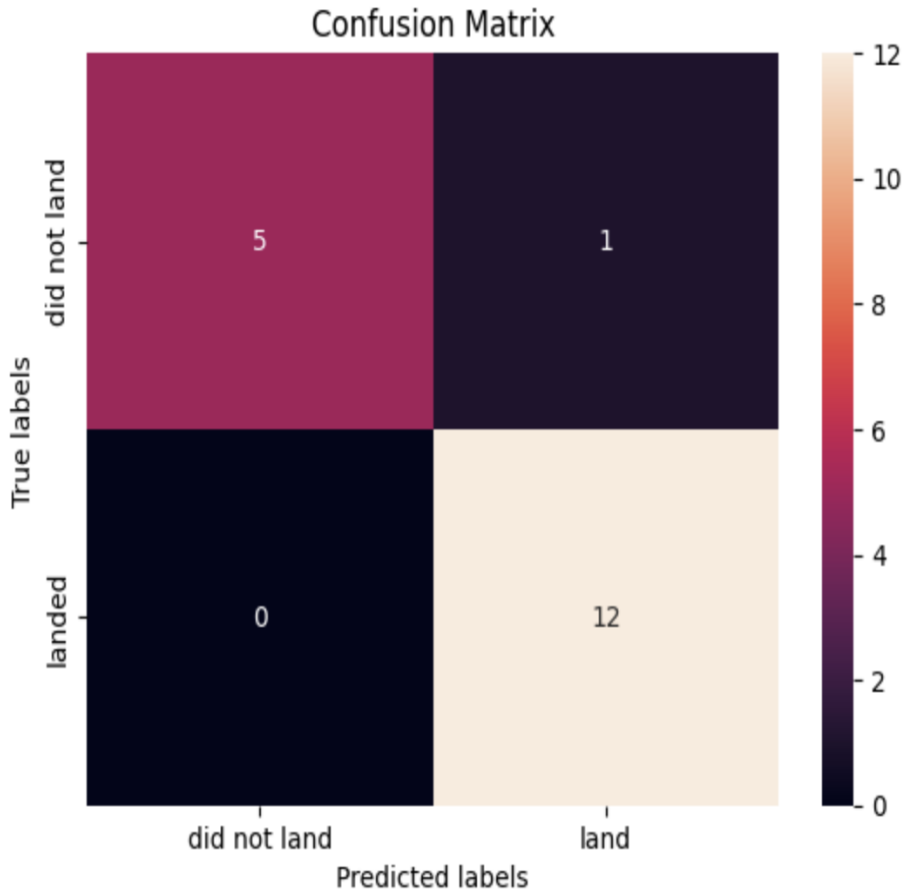
This image shows the Confusion Matrix for the K-Nearest Neighbor model.

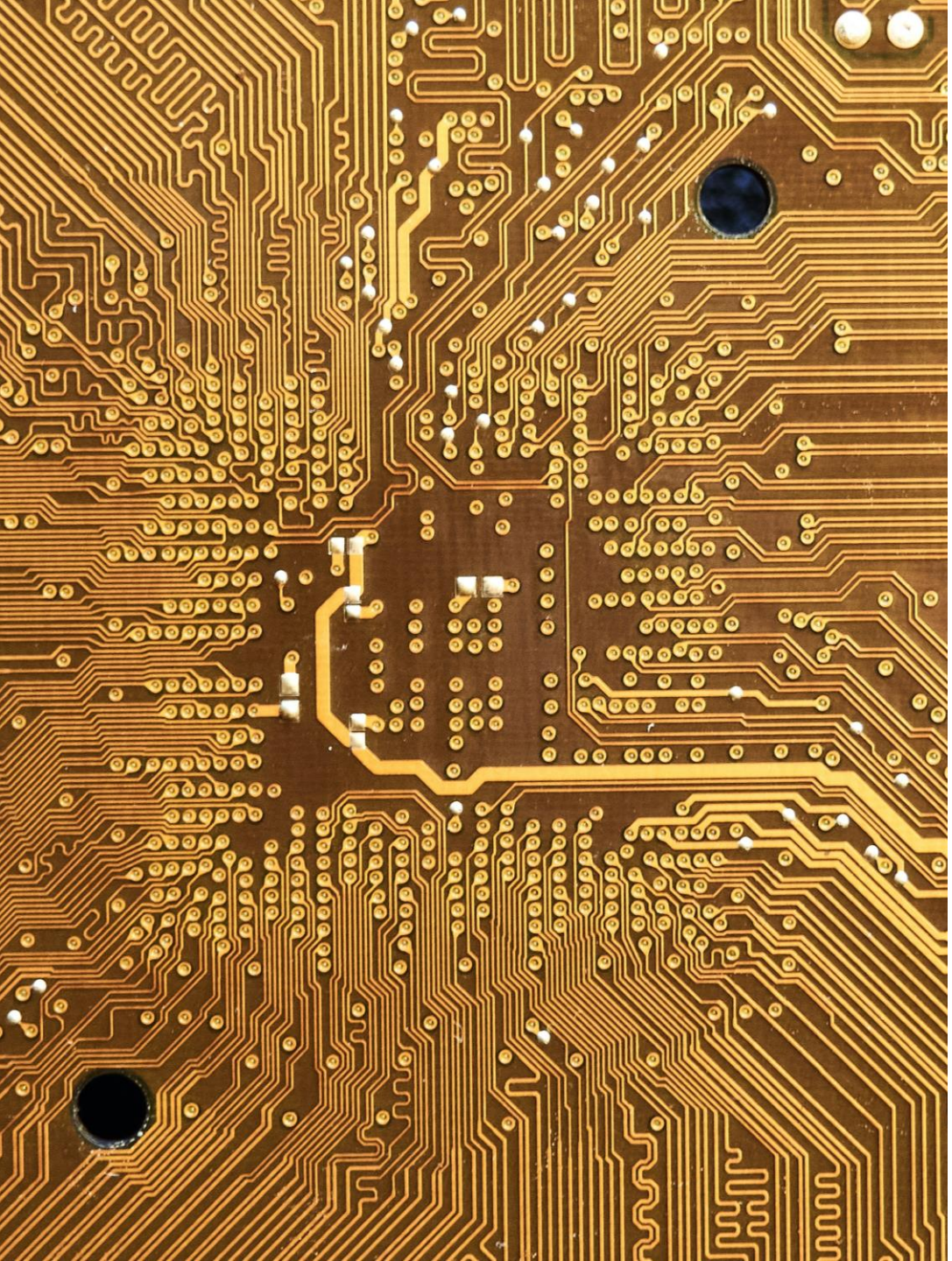
We can see that the K-NN machine learning classification model predicted successfully 5 landing failures and 12 successful landing, and only unsuccessfully predicted one failure to land and one successful landing.

We note that the Confusion Matrix outcomes are the same for SVM and Logistic Regression models, with the Decision Tree model the only model to score considerably worse.

KNN has been chosen on the basis of a broader evaluation of validation metrics.

```
[111]: # Make predictions on the knn test data
       yhat = knn_cv.predict(X_test)
       # Plot the confusion matrix
       plot_confusion_matrix(Y_test,yhat)
```





Conclusion

In conclusion, this project positions a theoretical company as a competitive and reliable player in the dynamic space launch industry. Leveraging machine learning driven insights, the company can make informed decisions across various product lines, enhancing its overall success and strategic positioning in a highly competitive market.

Data Collection used Python, Jupyter notebooks, Pandas, NumPy, and BeautifulSoup for API calls and web scraping, wrangling data, cleaning, pre-processing and feature engineering.

EDA and Visualization explored trends and success factors with SQL/SQLalchemy, visualizations, matplotlib, seaborn, Folium, and Plotly Dash.

Predictive Analysis: developed and validated machine learning classification models (selecting KNN) and optimized hyperparameters for risk, cost, and strategic planning, using above libraries and modules, as well as scikitlearn.

Key Findings: SpaceX exhibits consistent improvement in launch success rates, with KSC LC-39A standing out as the most successful launch site. The distribution of launches provides opportunities for targeted pricing strategies, based on product and location, and the machine learning model enhances precision in predicting success.

Strategic Insights: The machine learning analysis made possible will equip a theoretical company with actionable insights for improving cost estimation, competitive bidding, risk management, market differentiation, strategic planning, resource optimization, customer satisfaction, adaptability to market trends, and regulatory compliance.

Appendix

Additional Insights:

Success Factors and Trends: SpaceX shows consistent improvement in launch success rates over time, varying across orbit types, with ESL-1, GEO, HEO, and SSO having the highest success rates, noting that these rates are likely due to leveraging existing experimentation in other orbit classes.

Launch Site Analysis: KSC LC-39A stands out as the most successful launch site, benefiting from ocean proximity for safety measures and cost-effectiveness.

Pricing and Market Strategy: Distribution of launches across orbit and payload types offers opportunities for targeted pricing strategies in less competitive areas, enhancing competitiveness against industry leaders.

Machine Learning Predictions: The K-Nearest Neighbor model demonstrates promising accuracy for predicting successful landings, serving as a valuable tool for risk mitigation, cost estimation, and strategic planning. This can likely be further enhanced using ensemble approaches and gradient boosting algorithms.

Geographical Insights: Launch site distribution and proximity to the equator correlate with success, highlighted visually through Folium maps, offering logistical advantages. This can also inform bidding and pricing strategies in different markets.

Key Performance Indicators: Booster versions, payload mass, and launch outcomes are critical indicators, guiding strategic targeting for product development and pricing strategies.

Limitations and Further Analysis: Acknowledging dataset limitations, further analysis is recommended to explore failure rate reasons, especially during inflection points like 2018, and to investigate the significance of the spiral pattern in launch site locations.

Appendix continued

Additional Insights continued

Competitive Bidding: The precision of machine learning predictions will strengthen bidding effectiveness against SpaceX, enabling strategic pricing adjustments for enhanced competitiveness and profitability.

Risk Mitigation: Insight into landing probabilities facilitates a comprehensive assessment and mitigation of launch risks, thereby bolstering the company's reputation and reliability.

Market Differentiation: We can also use this type of machine learning to predict our own launch successes. Consistently achieving accurate predictions will serve as a distinctive competitive advantage, appealing to customers who prioritize predictability and cost-effectiveness.

Strategic Planning: Informed by machine learning predictions, strategic planning can focus on launches with higher success probabilities, tailored to specific conditions or regions.

Optimized Resource Allocation: Understanding success likelihood will allow companies to optimize resource allocation, channeling investments strategically toward launches with higher success rates.

Customer Trust: The accuracy of machine learning predictions will contribute significantly to customer trust and satisfaction, aligning with successful outcomes and fostering a positive client-provider relationship.

Adaptability to Market Trends: The model's flexibility in adapting to evolving market trends will ensures any company using the machine learning predictions will experience sustained competitiveness in a dynamic industry.

Regulatory Compliance: Reliable predictions play a pivotal role in ensuring regulatory compliance, guiding operational alignment with industry standards and minimizing environmental impact. These machine learning derived predictions can support effective regulatory compliance.

Thank you!