IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Rodney Litttlejohn
1/26/2025

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection through API
  - Data Collection with Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis with SQL
  - Exploratory Data Analysis with Data Visualization
  - Interactive Visual Analytics with Folium
  - Machine Learning Prediction
- Summary of all results
  - Exploratory Data Analysis result
  - Interactive Analytics
  - Predictive Analytics from Machine Learning Lab

Introduction

- SpaceX has revolutionized the space industry by drastically reducing the cost of rocket launches, with its Falcon 9 launches priced as low as $62 million—significantly lower than other providers, who charge upwards of $165 million per launch. A key factor in these savings is SpaceX's innovative approach of reusing the first stage of the rocket, which is recovered and refurbished for future missions. By continuously reusing the first stage, the cost per launch is expected to decrease even further. As a data scientist at a startup competing with SpaceX, the goal of this project is to develop a machine learning pipeline that predicts the outcome of the first stage landing on future launches. This predictive model is crucial for determining the optimal bid to compete with SpaceX in the rocket launch market.
- Problems for this scenario includes:
  - Determine the key variables that impact the outcome of the rocket's first-stage landing
  - Explore how each factor influences the likelihood of a successful landing.
  - Identify the ideal conditions that maximize the chances of a successful landing

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was gathered using the SpaceX REST API and web scraping from Wikipedia.
- Perform data wrangling
  - Categorical features from the data were processed using one-hot encoding.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

Data collection involves gathering and measuring information on specific variables within a structured system, allowing for the analysis of relevant questions and evaluation of outcomes. In this case, the dataset was obtained through a REST API and web scraping from Wikipedia.

For the REST API, the process began with a GET request. The response content was then decoded as JSON and converted into a Pandas DataFrame using json_normalize(). The data was subsequently cleaned, checked for missing values, and filled as needed.

For web scraping, BeautifulSoup was used to extract launch records from an HTML table. The table was then parsed and converted into a Pandas DataFrame for further analysis.

Data Collection – SpaceX API

- Get request for rocket launch data using SpaceX API
- Use json_normalize method to convert json result to dataframe
- Cleaned the data and filled in missing values

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

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

```
# Use json_normalize meethod to convert the json result into a dataframe
data = pd.json_normalize(response.json())
```

```
# Lets take a subset of our dataframe keeping only the features we want and the flight number, and date_utc.
data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc']]

# We will remove rows with multiple cores because those are falcon rockets with 2 extra rocket boosters and rows t
data = data[data['cores'].map(len)==1]
data = data[data['payloads'].map(len)==1]

# Since payloads and cores are lists of size 1 we will also extract the single value in the list and replace the f
data['cores'] = data['cores'].map(lambda x : x[0])
data['payloads'] = data['payloads'].map(lambda x : x[0])

# We also want to convert the date_utc to a datetime datatype and then extracting the date leaving the time
data['date'] = pd.to_datetime(data['date_utc']).dt.date

# Using the date we will restrict the dates of the launches
data = data[data['date'] <= datetime.date(2020, 11, 13)]
```

From:
https://github.com/rlitt579/Applied-Data-Science-Capstone-Space-X/blob/main/jupyter-labs-spacex-data-collection-api.ipynb

Data Collection - Scraping

- Request from URL the Falcon9 Launch Wiki page
- Create a BeautifulSoup from the HTML response
- Extract all column/variable names from the HTML header

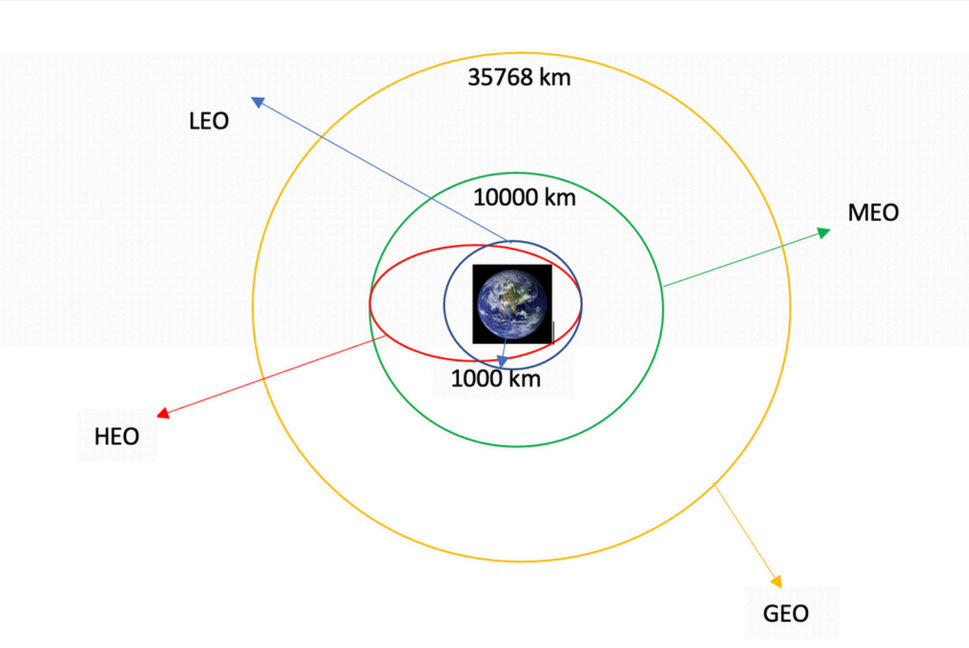
From: https://github.com/rlitt579/Applied-Data-Science-Capstone-Space-X/blob/main/jupyter-labs-webscraping.ipynb

```
]: # use requests.get() method with the provided static_url
   # assign the response to a object
   data = requests.get(static_url).text
```

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(data,'html.parser')
```

```
: extracted_row = 0
  #Extract each table
  for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowhea
     # get table row
      for rows in table.find_all("tr"):
          #check to see if first table heading is as number corresponding to launc
          if rows.th:
              if rows.th.string:
                  flight_number=rows.th.string.strip()
                  flag=flight_number.isdigit()
          else:
              flag=False
          #get table element
```

Data Wrangling

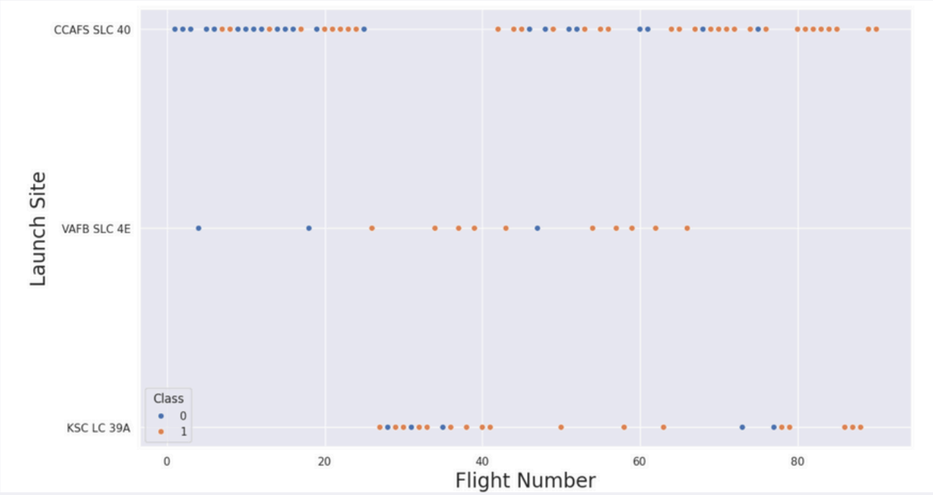


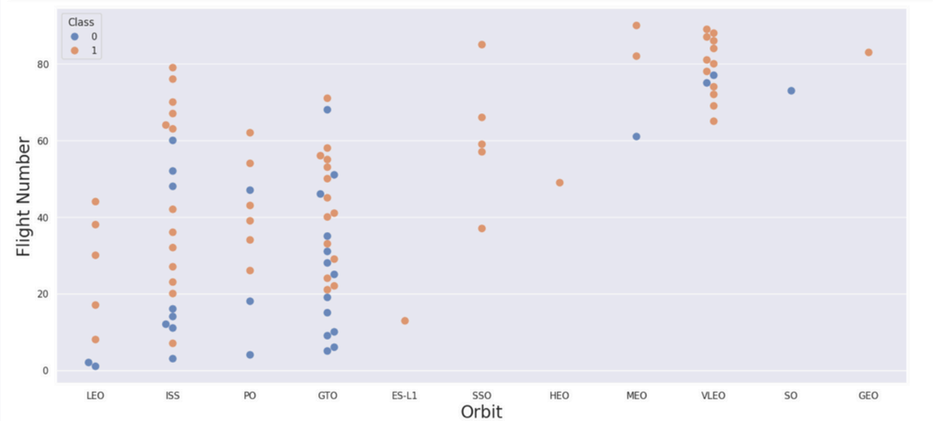
From:
https://github.com/rlitt579/Applied-Data-Science-Capstone-Space-X/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb

Data wrangling involves cleaning and organizing complex datasets to ensure easy access and effective exploratory data analysis (EDA).
My process includes:

- Calculating the number of launches at each site.
- Determining the frequency of mission outcomes for each orbit type.
- Creating a landing outcome label from the outcome column to simplify further analysis, visualization, and machine learning.
- Exporting the final results to a CSV file for future use.

EDA with Data Visualization





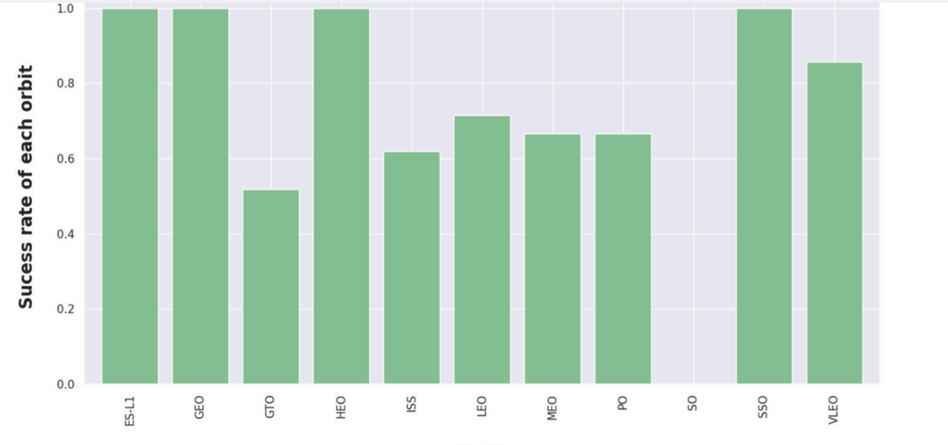
From: https://github.com/rlitt579/Applied-Data-Science-Capstone-Space-X/blob/main/edadataviz.ipynb

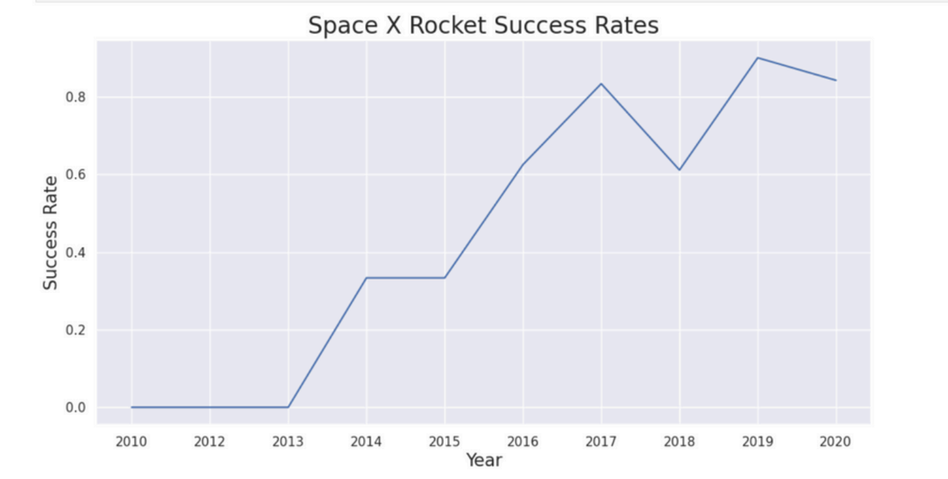
We began by utilizing scatter plots to explore relationships between various attributes, including:

- Payload vs. Flight Number
- Flight Number vs. Launch Site
- Payload vs. Launch Site
- Flight Number vs. Orbit Type
- Payload vs. Orbit Type

Scatter plots help visualize dependencies between attributes, allowing us to identify patterns. Once these patterns emerge, it becomes easier to determine which factors have the greatest influence on landing success.

EDA with Data Visualization





After gaining initial insights from scatter plots, we use additional visualization tools like bar graphs and line plots for deeper analysis.

- Bar Graphs: A simple and effective way to interpret relationships between attributes. Here, we use bar graphs to identify which orbits have the highest probability of success.
- Line Graphs: Used to observe trends over time, particularly to analyze yearly launch success patterns.

Finally, we apply Feature Engineering to enhance future success predictions by creating dummy variables for categorical columns, making them suitable for machine learning models.

From: https://github.com/rlitt579/Applied-Data-Science-Capstone-Space-X/blob/main/edadataviz.ipynb

EDA with SQL

Using SQL, I executed various queries to gain deeper insights into the dataset.
Examples include:

- Retrieving the names of all launch sites.
- Extracting five records where launch site names start with ‘CCA’.
- Calculating the total payload mass carried by boosters launched by NASA (CRS).
- Determining the average payload mass carried by the F9 v1.1 booster version.
- Identifying the date of the first successful landing outcome on a ground pad.
- Listing boosters that successfully landed on a drone ship and carried a payload mass between 4000 and 6000.
- Counting the total number of successful and failed mission outcomes.
- Identifying booster versions that carried the maximum payload mass.
- Retrieving failed landing outcomes on drone ships, along with their booster versions and launch sites, for the year 2015.
- Ranking the frequency of successful landing outcomes between June 4, 2010, and March 20, 2017, in descending order.

From: https://github.com/rlitt579/Applied-Data-Science-Capstone-Space-X/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

To create an interactive map visualizing the launch data, I utilized the latitude and longitude coordinates of each launch site and added circle markers labeled with their respective names.

- Next, we categorized the launch_outcomes data into two classes:
- Success (1) – Represented by green markers
- Failure (0) – Represented by red markers
- These markers were displayed using MarkerCluster() to enhance visualization.

Additionally, I applied Haversine's formula to calculate the distances between launch sites and key landmarks to answer questions such as:

- How close are the launch sites to railways, highways, and coastlines?
- How close are the launch sites to nearby cities?

From: https://github.com/rlitt579/Applied-Data-Science-Capstone-Space-X/blob/main/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

I developed an interactive dashboard using Plotly Dash, enabling users to explore and analyze the data dynamically.

- Pie Charts: Displaying the total number of launches at each launch site.
- Scatter Plot: Visualizing the relationship between launch outcomes and payload mass (kg) across different booster versions.

From: https://github.com/rlitt579/Applied-Data-Science-Capstone-Space-X/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

Model Building Process

- Load the dataset using NumPy and Pandas.
- Preprocess and transform the data, then split it into training and test sets.
- Select the appropriate machine learning algorithm based on the problem type.
- Configure GridSearchCV to optimize hyperparameters and fit the model to the dataset.

Model Evaluation

- Assess the accuracy of each model.
- Retrieve the optimized hyperparameters for different algorithms.
- Visualize performance using a confusion matrix.

Predictive Analysis (Classification)

Model Enhancement

- Apply Feature Engineering to optimize input data.
- Perform Algorithm Tuning to enhance performance.

Selecting the Best Model

- Identify the model with the highest accuracy score as the best-performing model.

From: https://github.com/rlitt579/Applied-Data-Science-Capstone-Space-X/blob/main/spacex_dash_app.py

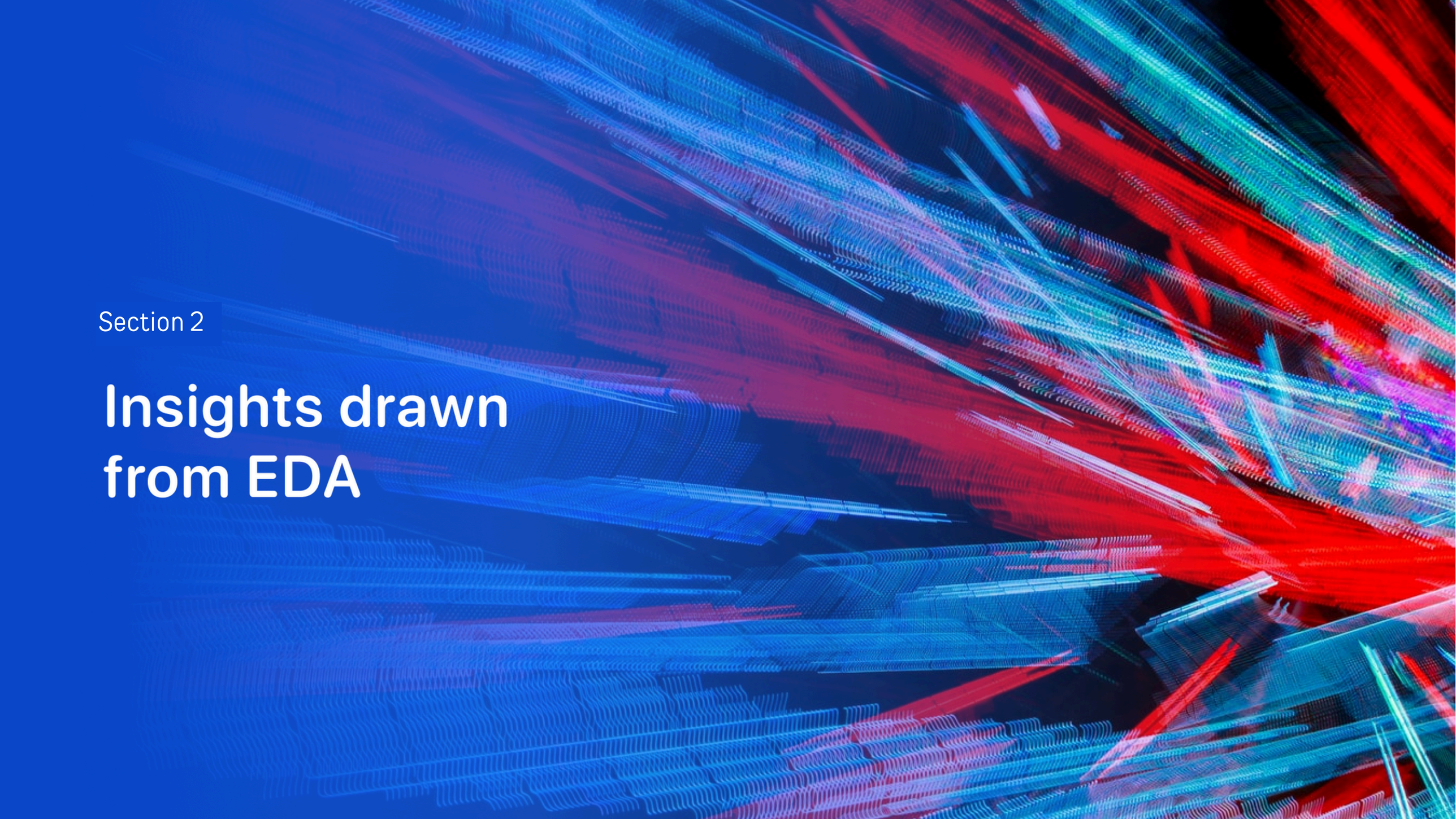
Results

The results are categorized into three main sections:

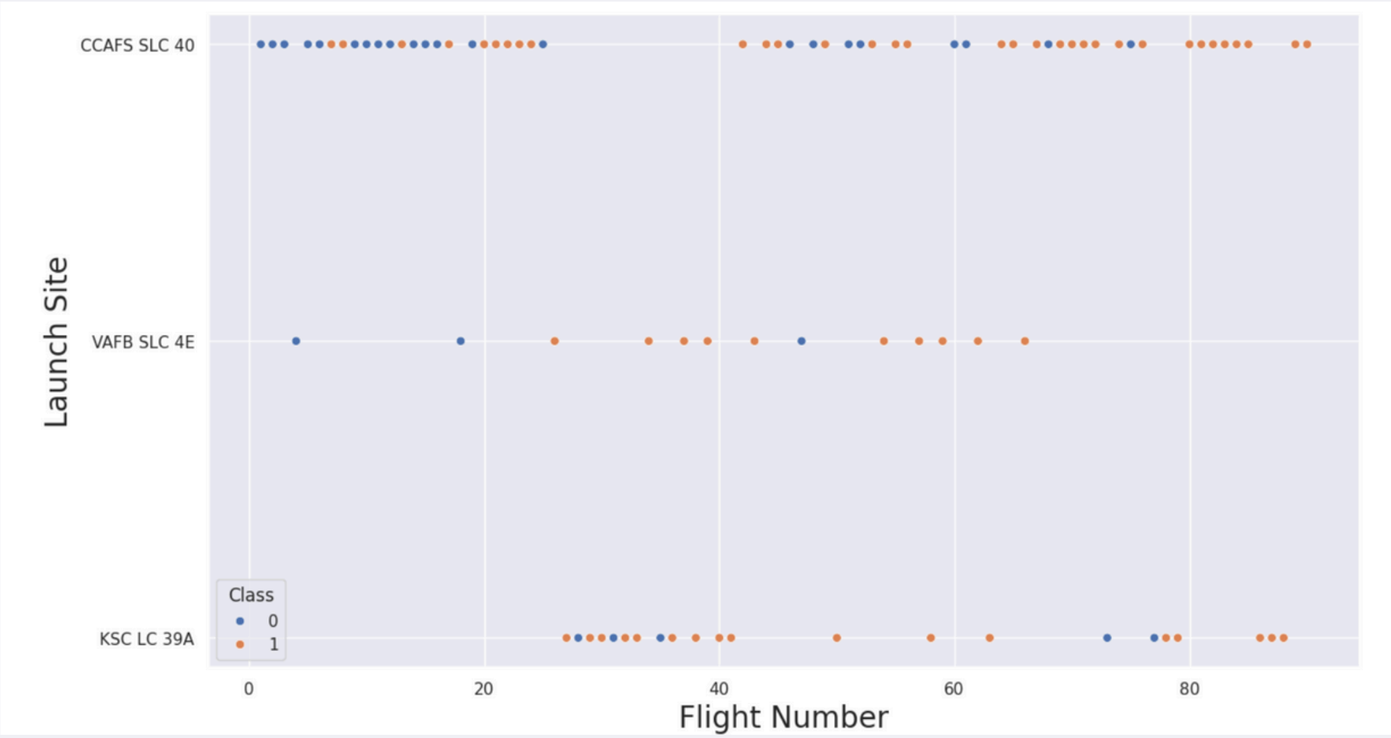
- Exploratory Data Analysis (EDA) Findings
- Interactive Analytics Demonstration (Screenshots)
- Predictive Analysis Outcomes

Section 2

Insights drawn from EDA



Flight Number vs. Launch Site



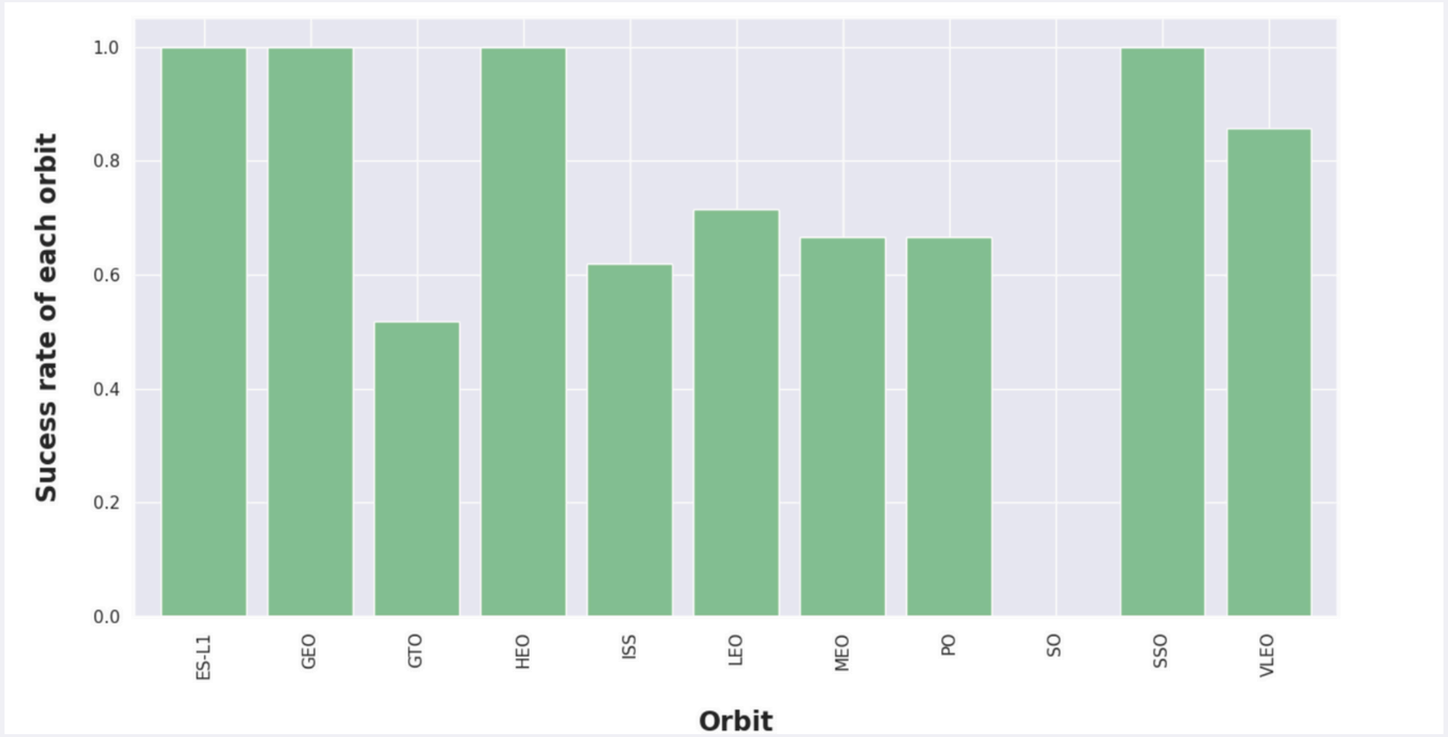
This scatter plot illustrates that as the number of flights from a launch site increases, the success rate tends to improve. However, CCAFS SLC-40 exhibits the weakest correlation with this trend.

Payload vs. Launch Site

This scatter plot indicates that when the payload mass exceeds 7000kg, the success rate significantly increases. However, there is no clear pattern suggesting that the launch site's success rate is directly influenced by payload mass

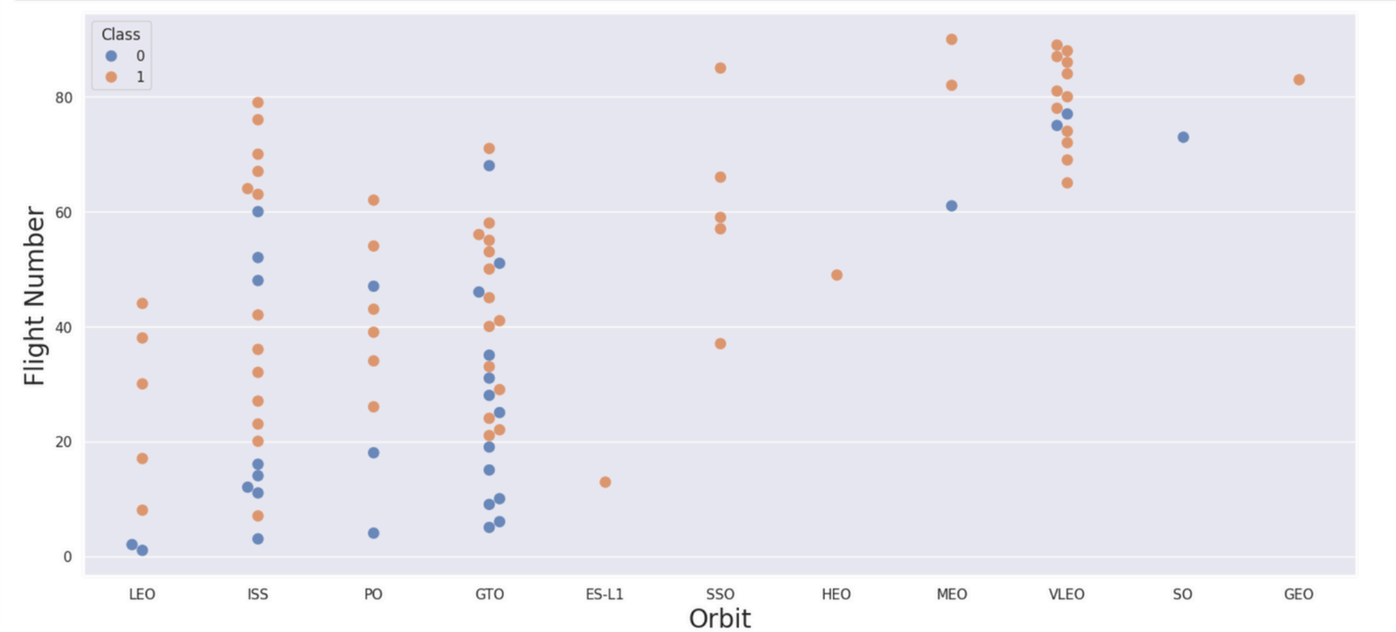
Success Rate vs. Orbit Type

This figure illustrates how orbital trajectories can influence landing outcomes. Certain orbits, like SSO, HEO, GEO, and ES-L1, exhibit a 100% success rate, while the SO orbit shows a 0% success rate. However, a more in-depth analysis reveals that some of these orbits, including GEO, SO, HEO, and ES-L1, only have a single occurrence, suggesting that more data is needed to identify patterns or trends before drawing any conclusions.



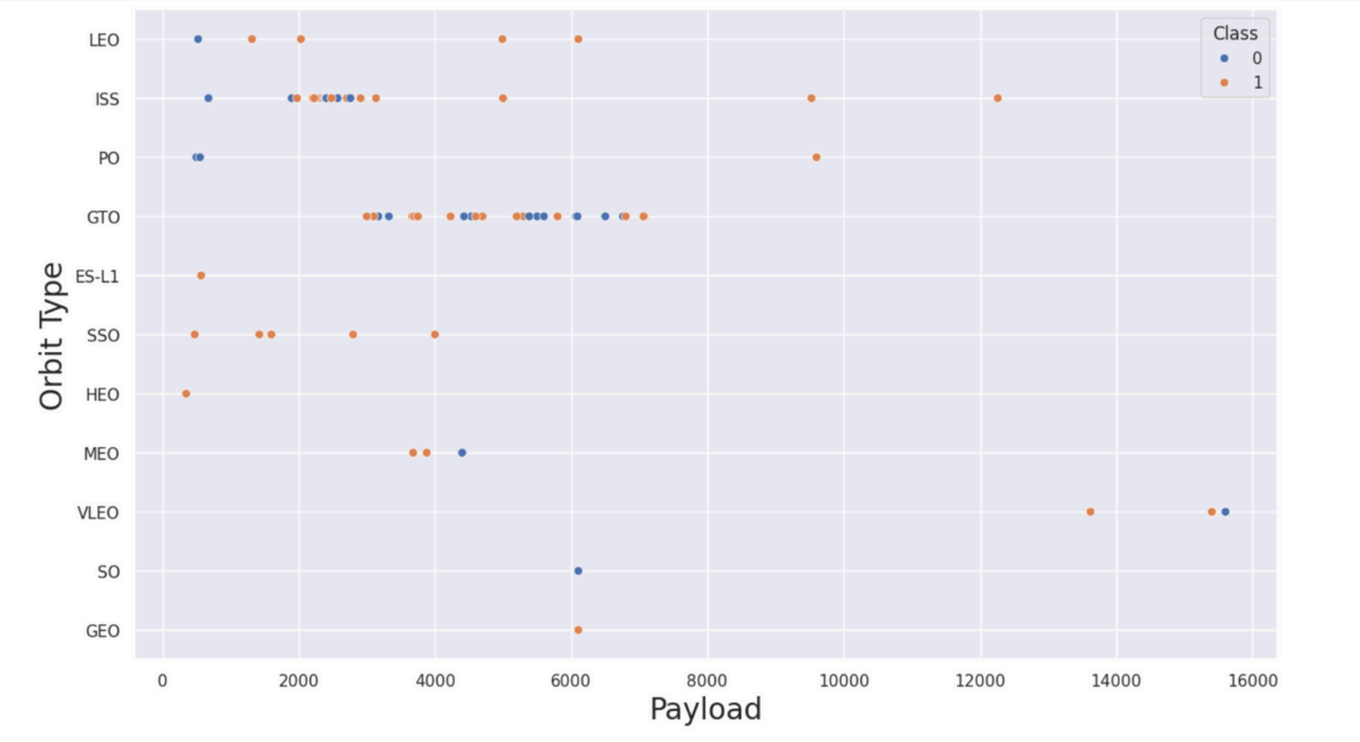
Flight Number vs. Orbit Type

This scatter plot indicates that, in general, a higher flight number for each orbit correlates with a greater success rate, particularly for the LEO orbit. However, the GTO orbit shows no apparent relationship between flight number and success rate. Orbits with only one occurrence should also be excluded from this observation, as additional data is needed for a more accurate analysis.



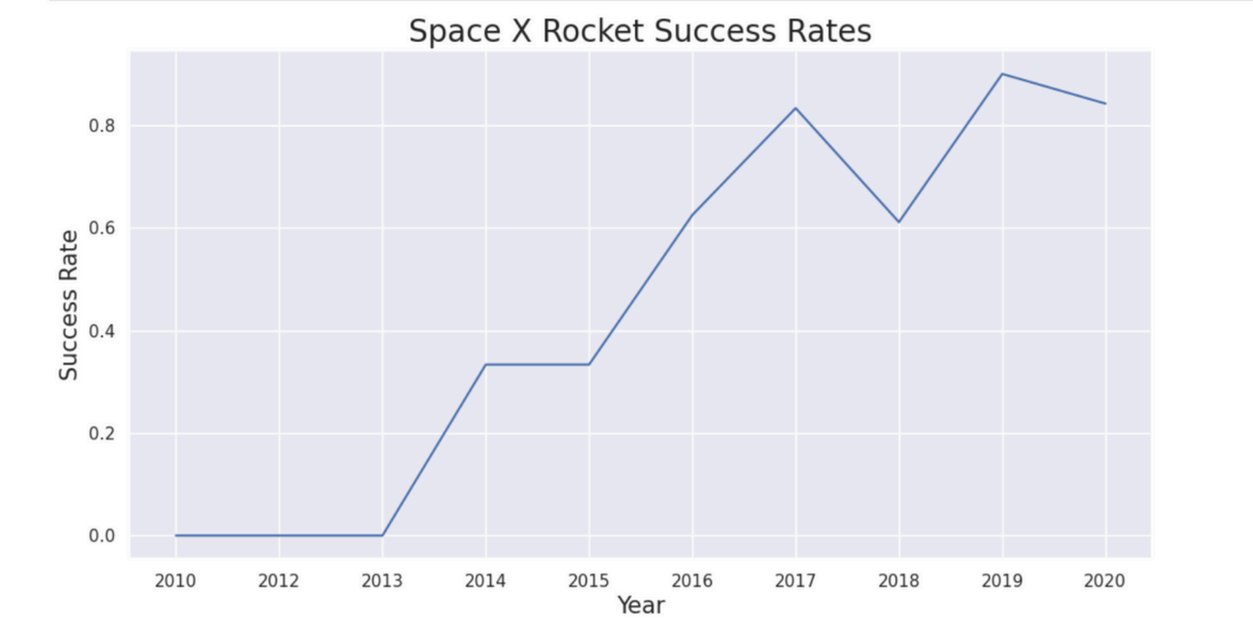
Payload vs. Orbit Type

A heavier payload has a positive impact on the LEO, ISS, and P0 orbits, but a negative impact on the MEO and VLEO orbits. The GTO orbit shows no apparent relationship between the payload and the success rate. Additionally, the SO, GEO, and HEO orbits require more data to identify any patterns or trends.



Launch Success Yearly Trend

This figure clearly shows an increasing trend from 2013 to 2020. If this trend continues in the coming years, the success rate will steadily rise, potentially reaching a 100% success rate.



All Launch Site Names

I used the keyword DISTINCT to display only the unique launch sites from the SpaceX data.

```
%sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL;
```

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

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

Launch Site Names Begin with 'CCA'

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

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

Launch_Site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

I used the query above to retrieve 5 records where the launch sites begin with "CCA."

Total Payload Mass

Using the query below, I calculated the total payload carried by NASA boosters as 45,596.

```
%sql SELECT SUM("PAYLOAD_MASS__kg_") AS "Total Payload Mass by NASA (CRS)" FROM SPACEXTBL WHERE "Customer" = 'NASA (CRS)';
```

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

Total Payload Mass by NASA (CRS)
45596

Average Payload Mass by F9 v1.1

- I calculated the average payload mass carried by the F9 v1.1 booster version as 2,928.4.

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS "Average Payload Mass by Booster" FROM SPACEXTBL WHERE BOOSTER_VERSION = 'F9 v1.1';
```

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

Average Payload Mass by Booster
2928.4

First Successful Ground Landing Date

I used the min() function to find the result. I observed that the date of the first successful landing outcome on the ground pad was December 22, 2015.

```
%sql SELECT MIN (DATE) AS "First Successful Landing" FROM SPACEXTBL WHERE LANDING_OUTCOME = 'Success (ground pad)';
```

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

First Successful Landing
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

I used the WHERE clause to filter for boosters that successfully landed on the drone ship and applied the AND condition to select those with a payload mass greater than 4000 but less than 6000.

```
%sql SELECT BOOSTER_VERSION FROM SPACEXTBL 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

I used the wildcard '%' to filter for where MissionOutcome was either a success or a failure.

```
%sql SELECT COUNT(MISSION_OUTCOME) AS "succesful mission "FROM SPACEXTBL WHERE MISSION_OUTCOME LIKE 'Success%'
```

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

succesful mission
100

```
%sql SELECT COUNT(MISSION_OUTCOME) AS "failure mission " FROM SPACEXTBL WHERE MISSION_OUTCOME LIKE 'Fail%'
```

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

failure mission
1

```
%sql SELECT sum(case when MISSION_OUTCOME LIKE '%Success%' then 1 else 0 end) AS "Successful Mission", \
    sum(case when MISSION_OUTCOME LIKE '%Failure%' then 1 else 0 end) AS "Failure Mission" \
FROM SPACEXTBL;
```

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

Successful Mission	Failure Mission
100	1

Boosters Carried Maximum Payload

- I determined the booster that carried the maximum payload using a subquery in the WHERE clause and the MAX() function.

```
%sql SELECT DISTINCT BOOSTER_VERSION AS "Booster Versions which carried the Maximum Payload Mass" FROM SPACEXTBL \
WHERE PAYLOAD_MASS__KG_ =(SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
```

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

Booster Versions which carried the Maximum Payload Mass
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

I used a combination of the WHERE clause, LIKE, AND, and BETWEEN conditions to filter for failed landing outcomes on the drone ship, along with their booster versions and launch site names for the year 2015.

```
%sql SELECT BOOSTER_VERSION, LAUNCH_SITE FROM SPACEXTBL WHERE DATE LIKE '2015-%' AND \
LANDING_OUTCOME = 'Failure (drone ship)';
```

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

Booster_Version	Launch_Site
F9 v1.1 B1012	CCAFS LC-40
F9 v1.1 B1015	CCAFS LC-40

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

I selected landing outcomes and the COUNT of landing outcomes from the data and used the WHERE clause to filter for landing outcomes between 2010-06-04 and 2010-03-20. I applied the GROUP BY clause to group the landing outcomes and the ORDER BY clause to order the grouped landing outcomes in descending order.

```
%sql SELECT LANDING_OUTCOME as "Landing Outcome", COUNT(LANDING_OUTCOME) AS "Total Count" FROM SPACEXTBL \
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' \
GROUP BY  LANDING_OUTCOME \
ORDER BY COUNT(LANDING_OUTCOME) DESC ;
```

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

Landing Outcome	Total 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

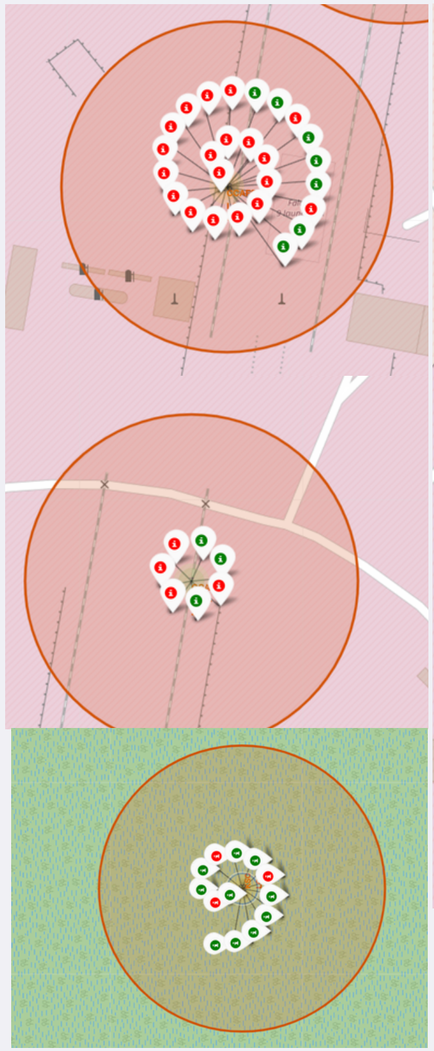
Section 3

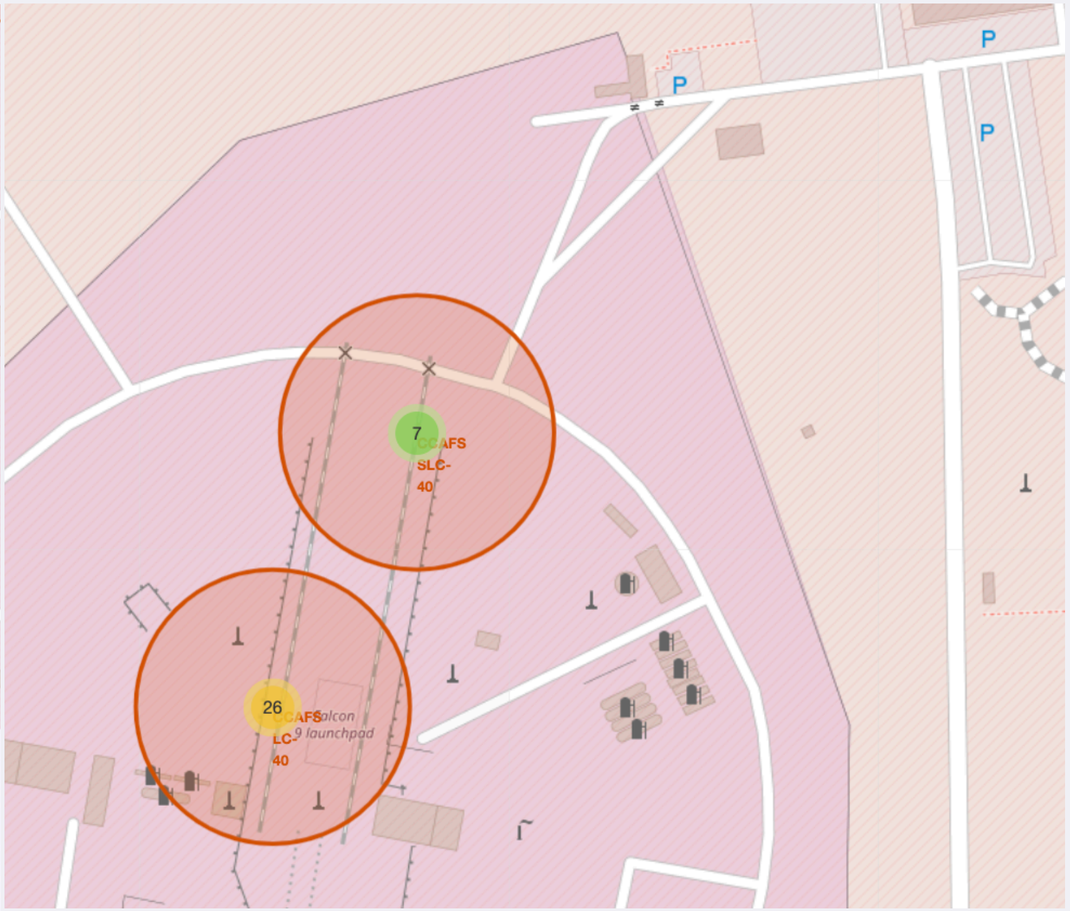
Launch Sites Proximities Analysis

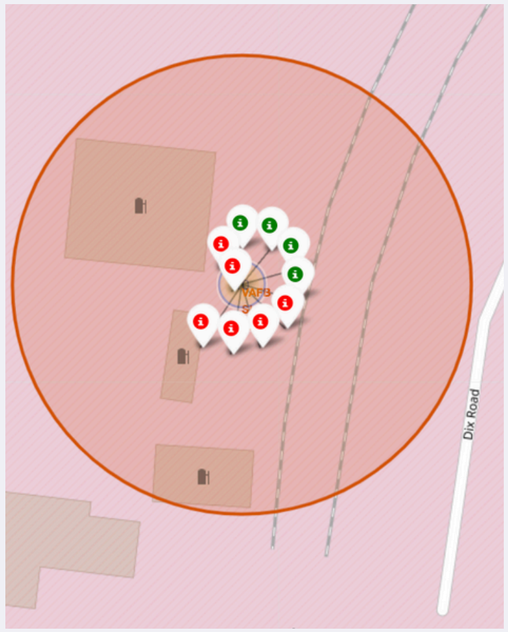
Launch Site Locations

It is evident that all the SpaceX launch sites are located within the United States.

Launch Sites with Color-Labeled Markers





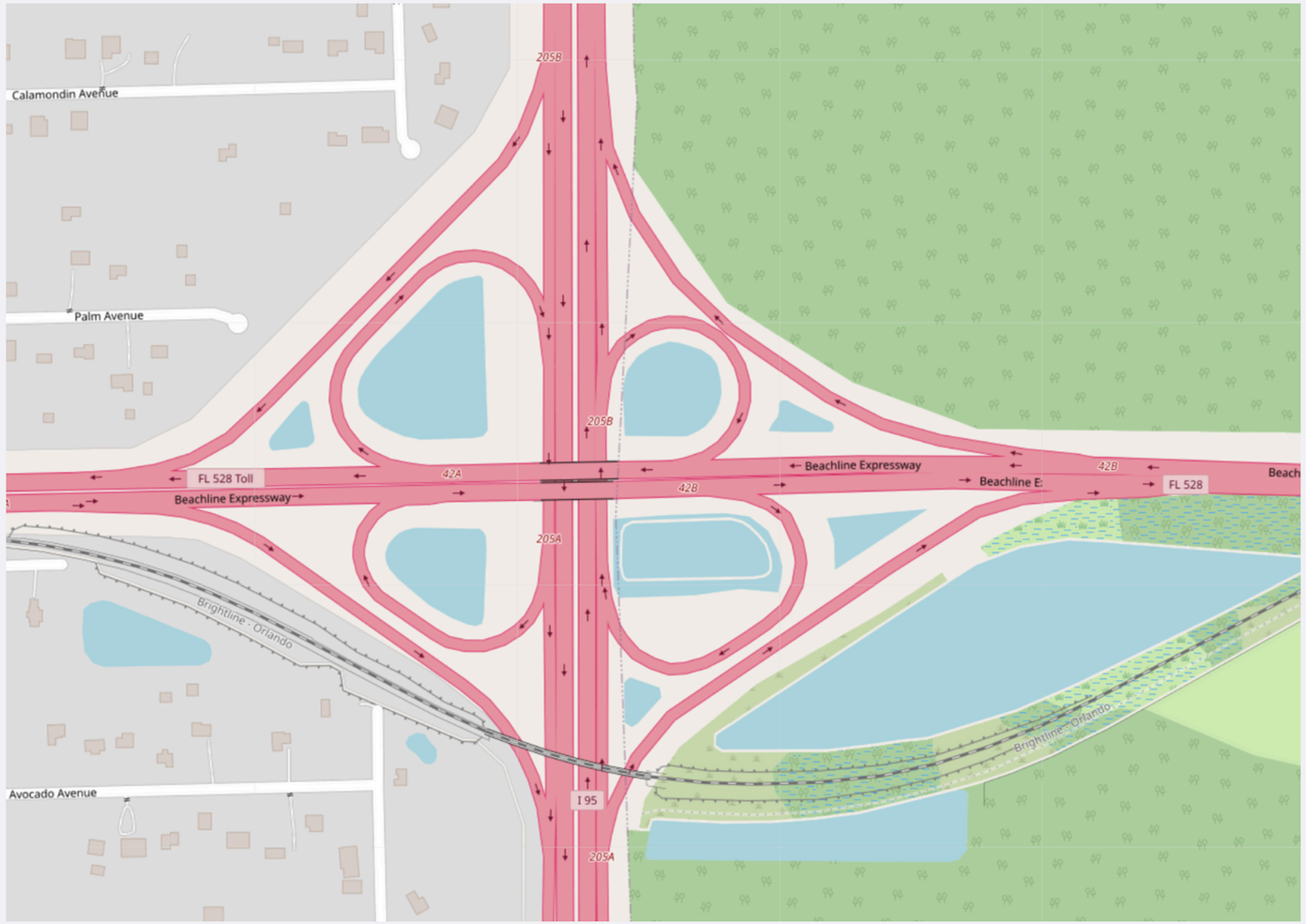


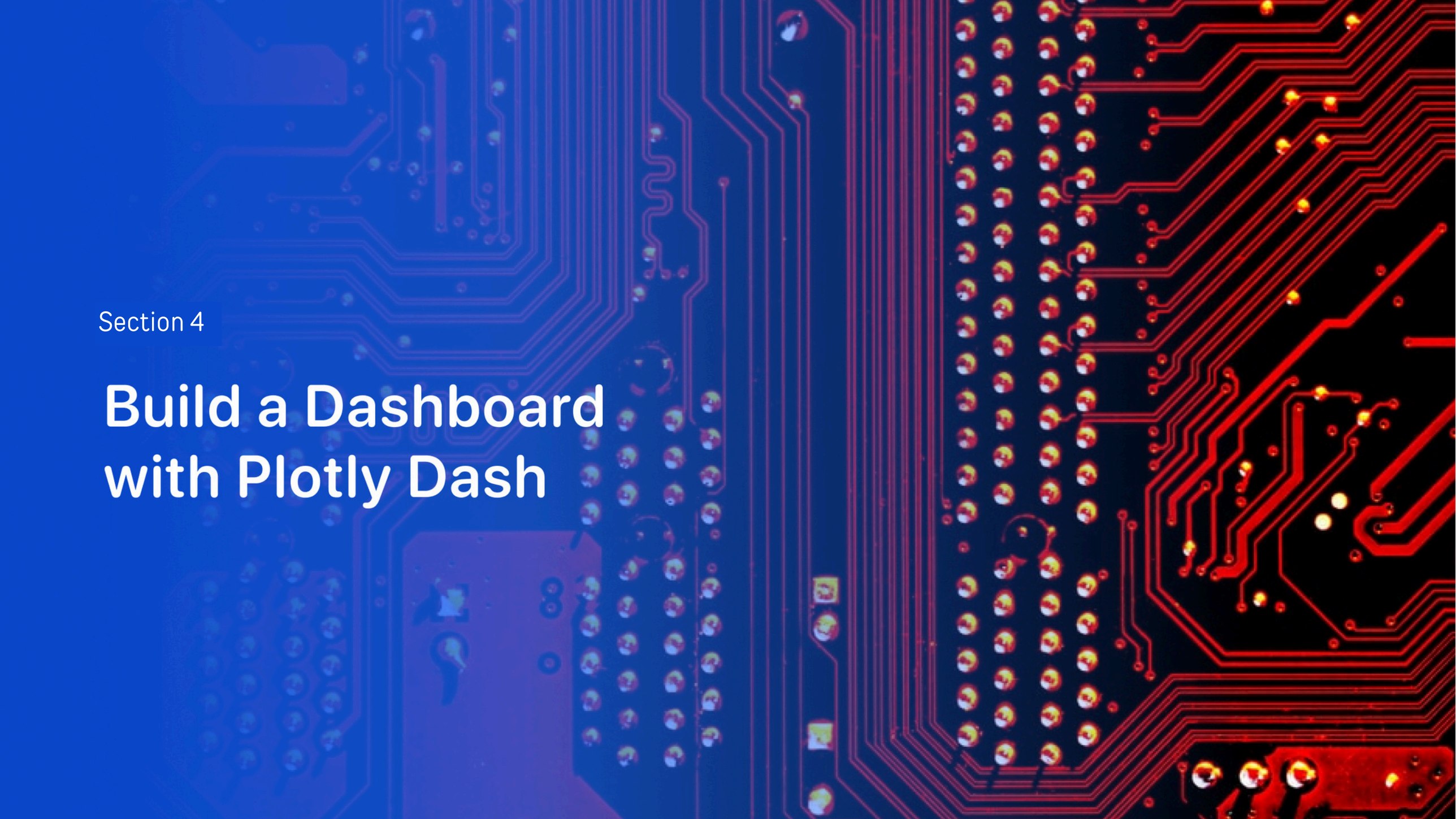
California Launch Site

Florida Launch Sites

Green Marker shows Successful Launches whereas Red Marker shows Failues

Launch Sites Distance to Landmarks





Section 4

Build a Dashboard with Plotly Dash

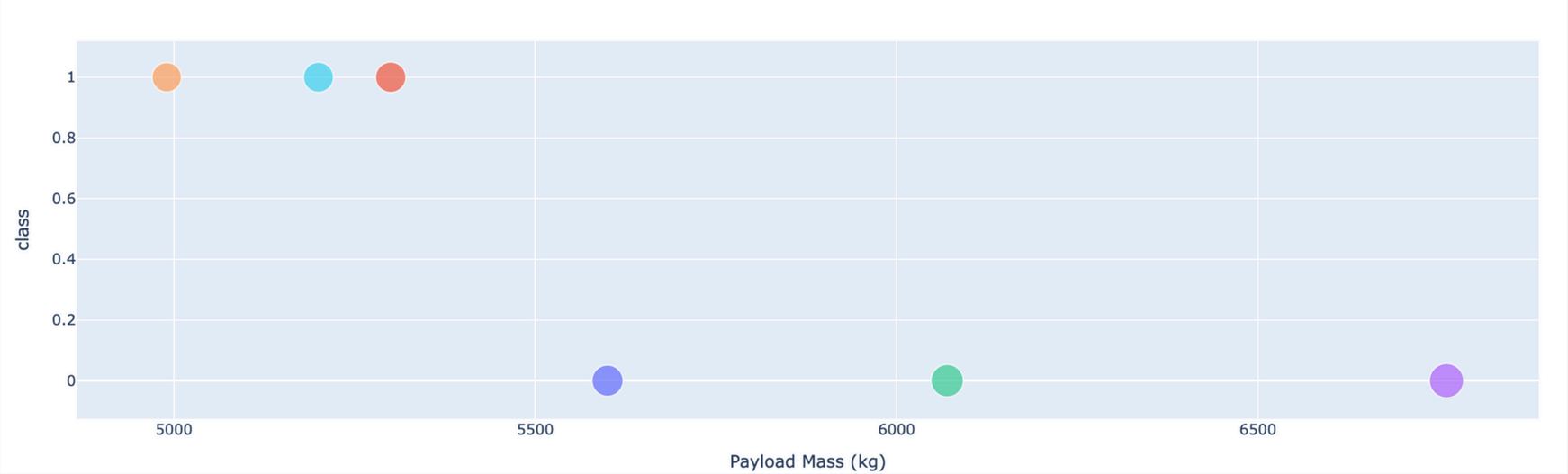
Success % by each Sites

Out of all of the sites, KSC LC-39A had the most successful launches.

Highest Launch-Success Ratio: KSC LC-39A

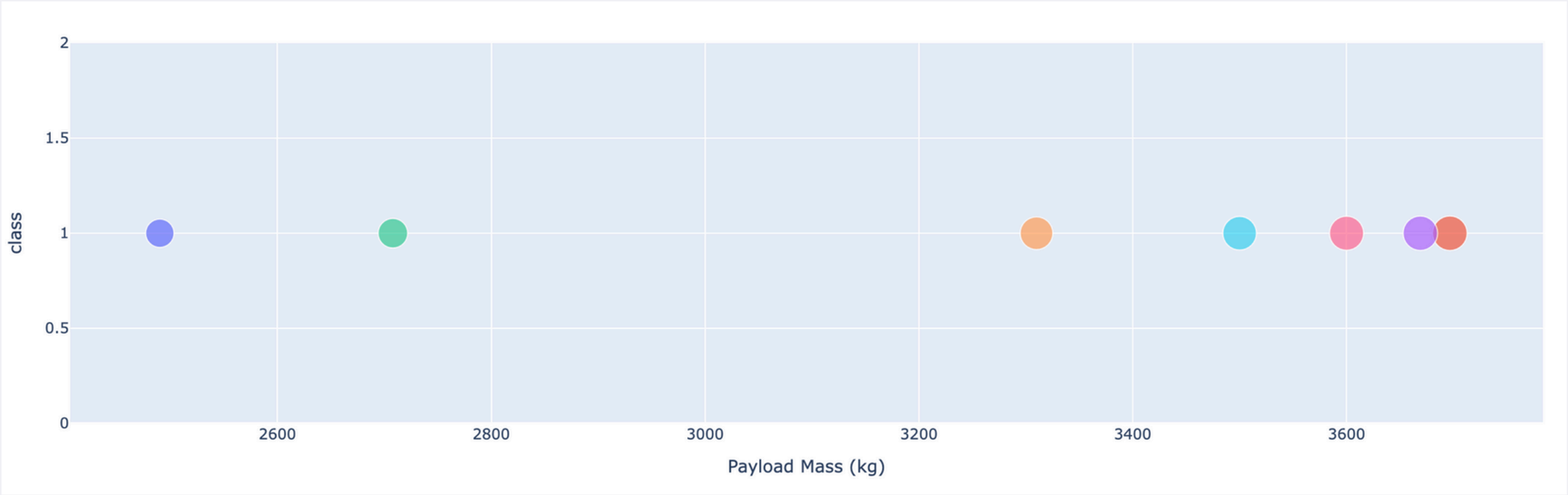
- ***KSC LC-39A achieved a 76.9% success rate with a 23.1% failure rate***

<Dashboard Screenshot 3>



The success rate for low weighted payload is higher than heavy weighted payload

Low Weighted Payload (0-4000kg)



Heavy Weighted Payload (4000-10000kg)

Section 5

Predictive Analysis (Classification)

Classification Accuracy

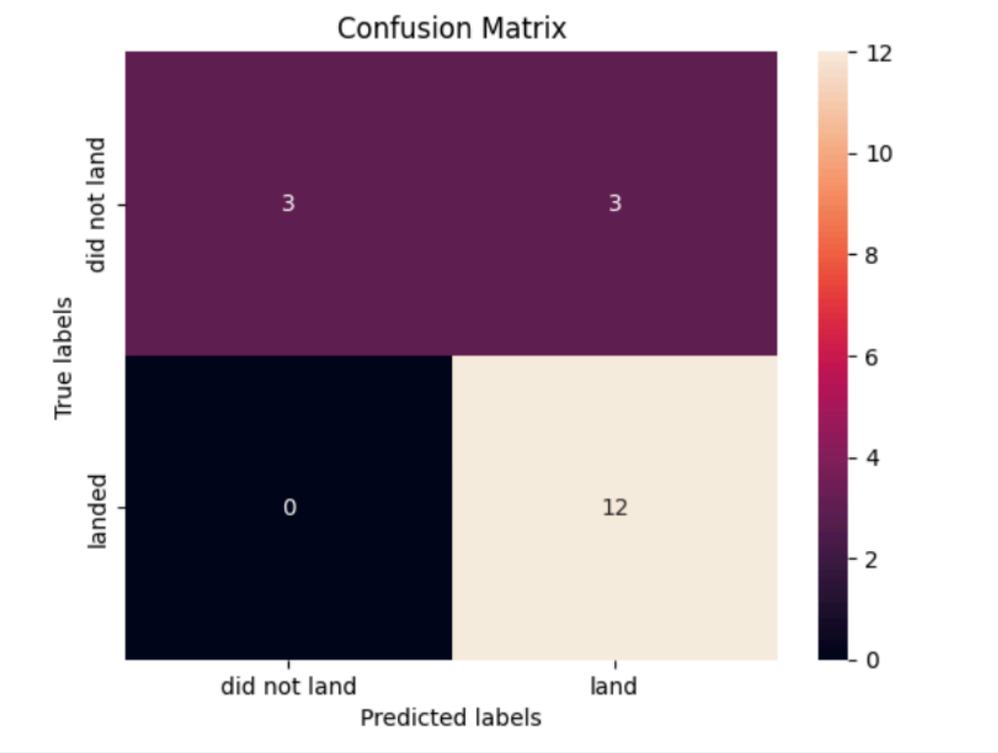
As shown in the code below, I identified the Tree Algorithm as the best algorithm, as it has the highest classification accuracy.

```
algorithms = {'KNN':knn_cv.best_score_,'Tree':tree_cv.best_score_,'LogisticRegression':logreg_cv.best_score_}
bestalgorithm = max(algorithms, key=algorithms.get)
print('Best Algorithm is',bestalgorithm,'with a score of',algorithms[bestalgorithm])
if bestalgorithm == 'Tree':
    print('Best Params is :',tree_cv.best_params_)
if bestalgorithm == 'KNN':
    print('Best Params is :',knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best Params is :',logreg_cv.best_params_)
```

```
Best Algorithm is Tree with a score of 0.875
Best Params is : {'criterion': 'entropy', 'max_depth': 4, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_samples_split': 5, 'splitter': 'random'}
```

Confusion Matrix

The confusion matrix for the decision tree classifier indicates that it can differentiate between the various classes. However, the main issue is the false positives, where the classifier incorrectly labels unsuccessful landings as successful.



Conclusions

Based on the analysis, I conclude that:

- The Tree Classifier Algorithm is the most effective machine learning approach for this dataset.
- Lower-weight payloads (4,000 kg and below) performed better than heavier payloads.
- Since 2013, the success rate of SpaceX launches has steadily increased, showing a direct correlation with time. This trend suggests that future launches may achieve near-perfect success rates.
- KSC LC-39A has the highest success rate among all launch sites, with 76.9% successful launches.
- The SSO orbit has the highest success rate at 100%, with more than one occurrence.

Appendix

- https://github.com/rlitt579/Applied-Data-Science-Capstone-Space-X

Thank you!