Winning Space Race with Data Science

Valentina Casadei
01/18/2024

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 in screenshots
  - Predictive Analytics result from Machine Learning Lab

Introduction

SpaceX, a ground-breaking company, has revolutionized the space industry by providing rocket launches, specifically the Falcon 9, at a significantly lower cost of 62 million dollars compared to other providers, whose prices soar to upwards of 165 million dollars each. This cost reduction is largely attributed to SpaceX's innovative approach of reusing the first stage of the launch by successfully re-landing the rocket for subsequent missions, ultimately driving prices even lower through repetitive use.
The primary objective of this project is to develop a machine learning pipeline that can accurately predict the landing outcome of the first stage in future missions. This undertaking is pivotal in determining the optimal bid price against SpaceX for a rocket launch.

Key challenges in this project include:

•Identifying all the factors that exert influence on the landing outcome.
•Understanding the relationships between these variables and their impact on the overall outcome.
•Determining the optimal conditions required to maximize the probability of a successful landing.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data collected using SpaceX REST API and web scrapping from Wikipedia
- Perform data wrangling
  - Data was processed using one-hot encoding for categorical features
- 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 is the systematic gathering of information on specific variables within a system to answer questions and evaluate outcomes. The dataset here was obtained through REST API and web scraping from Wikipedia.

- For REST API, a get request is used to collect data, which is then decoded as JSON and converted into a Pandas dataframe using json_normalize(). The data is cleaned, and missing values are filled.

- Web scraping involves using BeautifulSoup to extract launch records presented as an HTML table. The table is parsed, and the information is transformed into a Pandas dataframe for further analysis.

Data Collection – SpaceX API

Get request for the rocket launch data using API

Use json.normalize() method to convert the json file to dataframe

Data cleaning and filling missing values

From: https://github.com/BibiPhd/Data-Capstone---Coursera-/blob/main/Data%20Collection.ipynb

```
import requests
import pandas as pd
import numpy as np
import datetime

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

```
# JSON file
response = requests.get(space_y_url)
# print(response.content)
```

```
data_response = response.json()
data = pd.json_normalize(data_response)

# 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']]
#data_df.head()

# We will remove rows with multiple cores because those are falcon rockets with 2 extra rocket boosters and rows that have multip
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 feature.
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)]
```

Data Collection - Scraping

Request the Falcon9 Launch Wiki page from url and create BeautifulSoup from HTML response

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

```
soup = BeautifulSoup(response.content, 'html.parser')
```

Extract all columns or variables names from the HTML header

```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowheaders collapsible")):
   # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number corresponding to launch a number
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
        else:
            flag=False
        #get table element
        row=rows.find_all('td')
        #if it is number save cells in a dictonary
        if flag:
            extracted_row += 1
            # Flight Number value
            # TODO: Append the flight_number into launch_dict with key `Flight No.`
            #print(flight_number)
            datatimelist=date_time(row[0])

            # Date value
            # TODO: Append the date into launch_dict with key `Date`
            date = datatimelist[0].strip(',')
            #print(date)

            # Time value
            # TODO: Append the time into launch_dict with key `Time`
            time = datatimelist[1]
            #print(time)
```

From: https://github.com/BibiPhd/Data-Capstone---Coursera-/blob/main/Data%20Collection%20with%20web%20scraping.ipynb

Data Wrangling

Data wrangling is the procedure of cleaning and consolidating disorderly and intricate datasets to facilitate access and exploratory data analysis (EDA).

Initially, we will determine the count of launches on each site and calculate the number and frequency of mission outcomes per orbit type.

Subsequently, a landing outcome label will be generated from the outcome column, streamlining further analysis, visualization, and machine learning (ML).

Finally, the results will be exported to a CSV file.

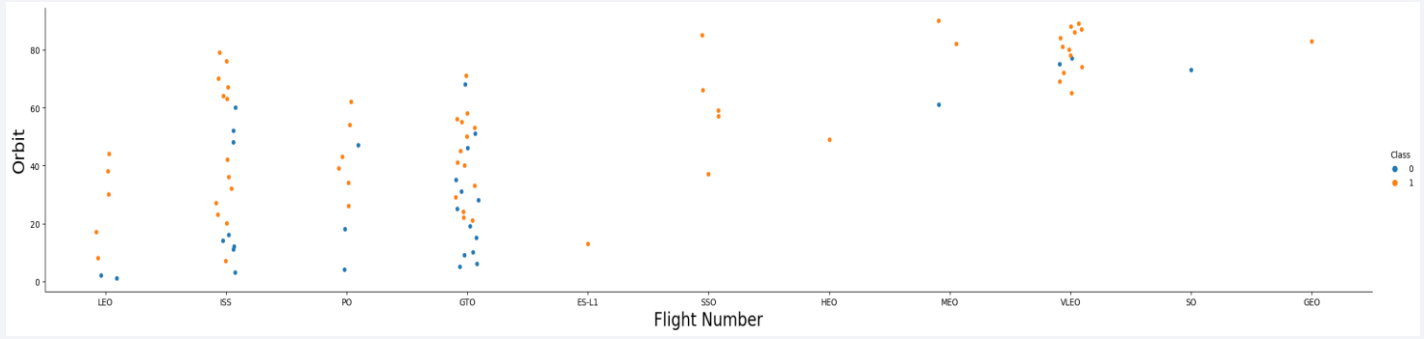
From: https://github.com/BibiPhd/Data-Capstone---Coursera-/blob/main/Data%20wrangling.ipynb

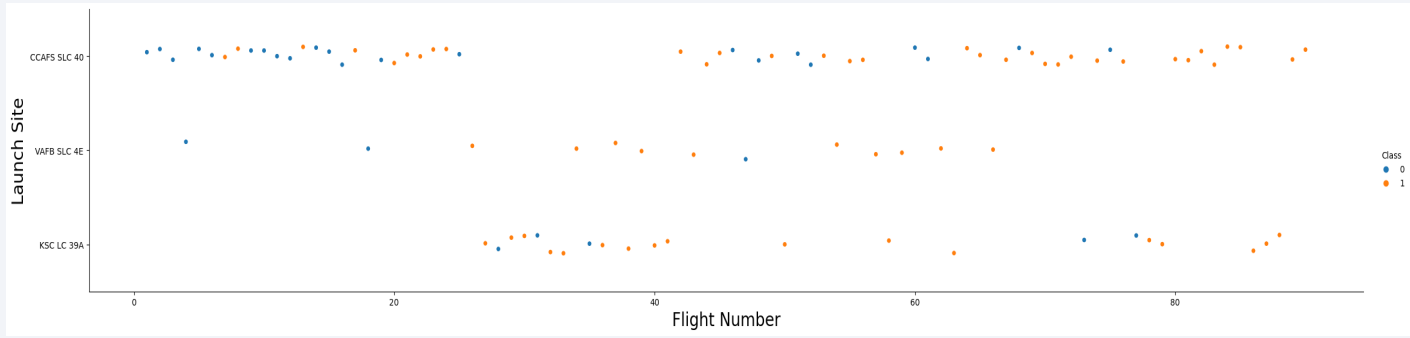
EDA with Data Visualization

Initially, we employed scatter graphs to check relationships among various attributes, including:

- Payload and Flight Number
- Flight Number and Launch Site
- Payload and Launch Site
- Flight Number and Orbit Type
- Payload and Orbit Type

Scatter plots effectively illustrate the dependencies between these attributes.

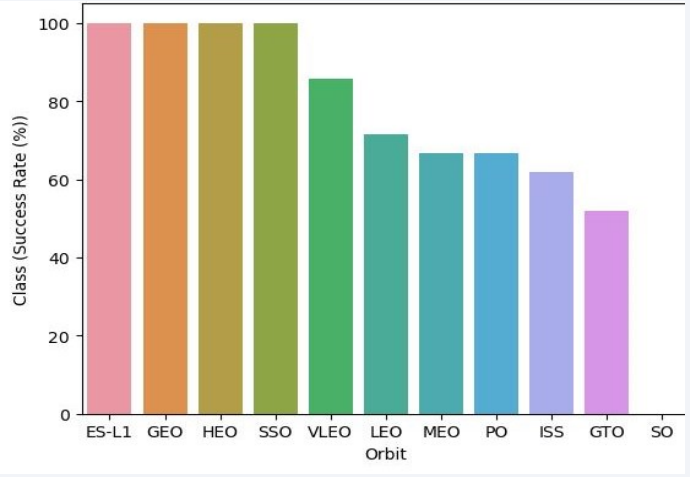




Analyzing the patterns identified in the graphs enables a clear understanding of the factors that exert the most influence on the success of landing outcomes.

From: https://github.com/BibiPhd/Data-Capstone---Coursera-/blob/main/Data%20analysis%20with%20visualization.ipynb

EDA with Data Visualization



After obtaining insights from the relationships identified through scatter plots, we extend our analysis using additional visualization tools, such as bar graphs and line plots.

Bar graphs offer a straightforward method to interpret relationships between attributes. In this instance, we utilize sorted bar graph to assess the orbits with the highest probability of success.

Subsequently, a line graph is employed to illustrate trends or patterns of attributes over time. In this case, it helps visualize the annual trend of launch success.

To enhance future success prediction modules, Feature Engineering is applied by creating dummy variables for categorical columns.

From: https://github.com/BibiPhd/Data-Capstone---Coursera-/blob/main/Data%20analysis%20with%20visualization.ipynb

EDA with SQL

Using SQL, we executed several queries to enhance our understanding of the dataset. Examples include:

- Displaying the names of the launch sites.
- Showing 5 records where launch sites start with the string 'CCA.'
- Revealing the total payload mass carried by boosters launched by NASA (CRS).
- Highlighting the average payload mass carried by booster version F9 v1.1.
- Listing the date of the first successful landing outcome on a ground pad.
- Enumerating the names of boosters with success in drone ship missions and a payload mass greater than 4000 but less than 6000.
- Summarizing the total number of successful and failed mission outcomes.
- Listing the names of booster versions carrying the maximum payload mass.
- Identifying failed landing outcomes on a drone ship, along with their booster versions and launch site names, for the year 2015.
- Ranking the count of landing outcomes or success between the dates 2010-06-04 and 2017-03-20 in descending order.

From: https://github.com/BibiPhd/Data-Capstone---Coursera-/blob/main/EDA%20with%20SQL.ipynb

Build an Interactive Map with Folium

To create an interactive map visualizing launch data, we utilized latitude and longitude coordinates for each launch site. Circle markers were added around each launch site, featuring a label indicating the site's name.

The launch outcomes dataframe (categorized as failure and success) was then assigned to classes 0 and 1, represented by Red and Green markers on the map using MarkerCluster().

To address questions related to proximity, we employed Haversine's formula to calculate the distances between launch sites and various landmarks, providing insights into:

•The proximity of launch sites to railways, highways, and coastlines.

•The closeness of launch sites to nearby cities.

From: https://github.com/BibiPhd/Data-Capstone---Coursera-/blob/main/Visual%20analytics%20with%20Folium.ipynb

Build a Dashboard with Plotly Dash

We developed an interactive dashboard using Plotly Dash, providing users the flexibility to explore the data interactively.

The dashboard includes:

•Pie charts displaying the total launches by specific sites.

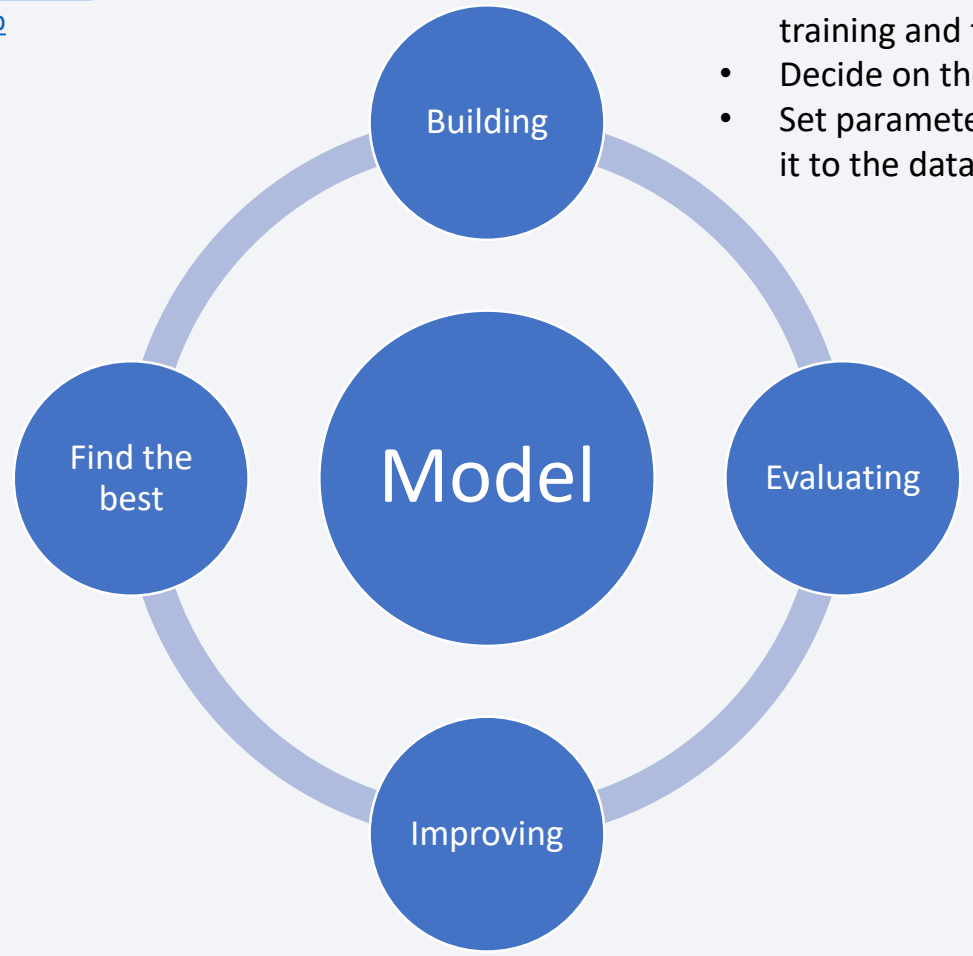
•Scatter graphs illustrating the relationship between Outcome and Payload Mass (Kg) for different booster versions. Users can manipulate and interact with these visualizations as per their requirements.

From: https://github.com/BibiPhd/Data-Capstone---Coursera-/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

From: https://github.com/BibiPhd/Data-Capstone---Coursera-/blob/main/Machine%20Learning%20Prediction.ipynb

- Load the dataset into NumPy and Pandas to be split into training and test datasets.
- Decide on the type of machine learning to use.
- Set parameters and algorithms for GridSearchCV, then fit it to the dataset.



- Check the accuracy for each model.
- Obtain tuned hyperparameters for each type of algorithm.
- Plot the confusion matrix.

The model with the best accuracy is the best one

With Feature Engineering and Algorithm Tuning

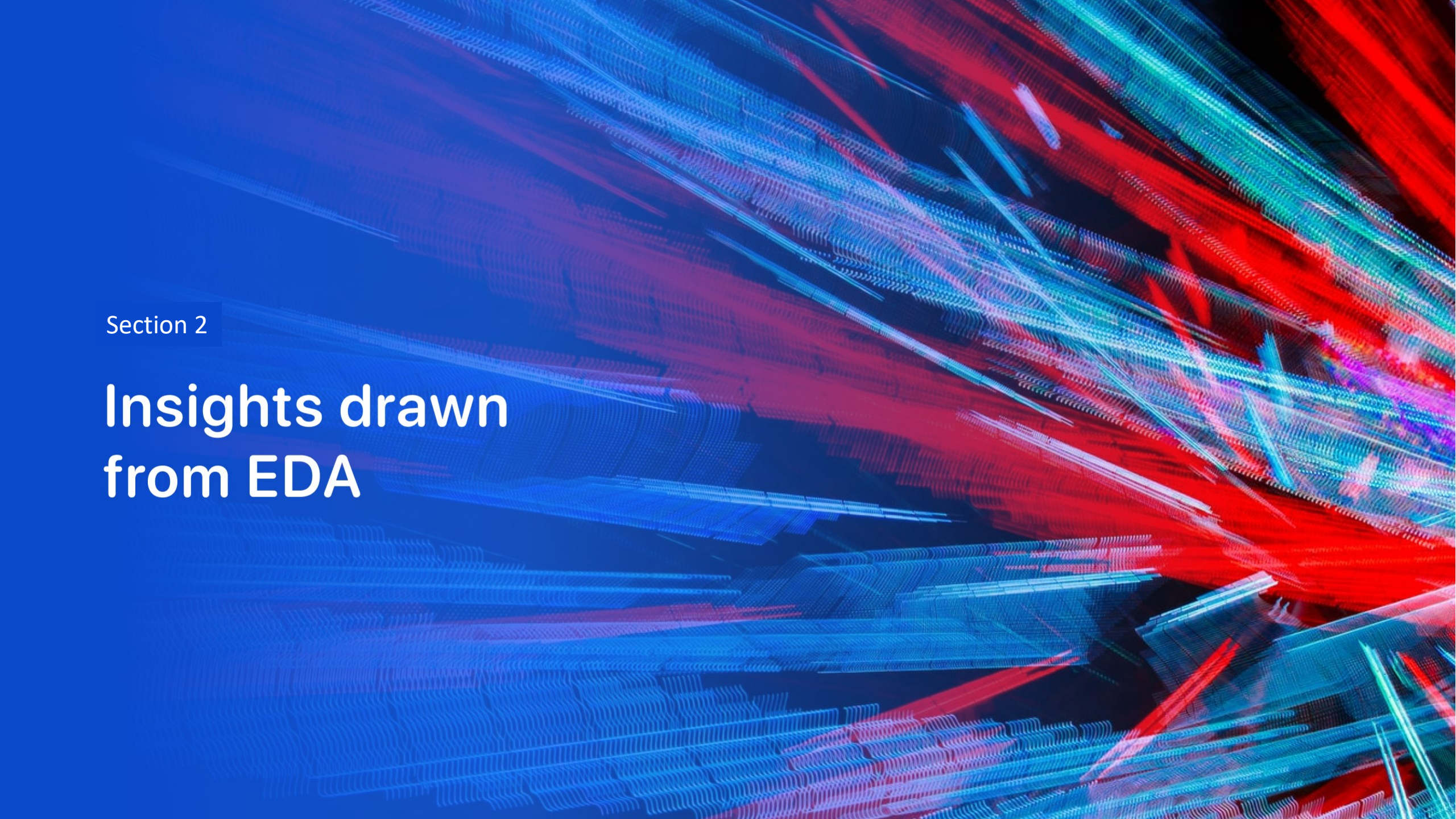
Results

The outcomes will be classified into three main categories:

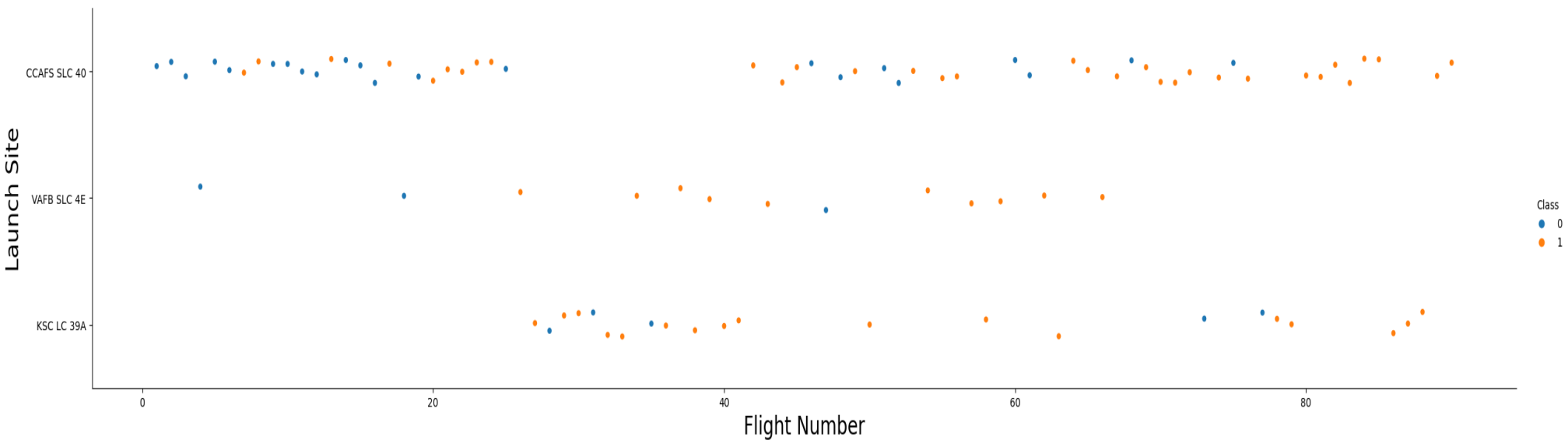
1. Exploratory Data Analysis Results
2. Screenshots of the Interactive Analytics Demo
3. Predictive Analysis Results

Section 2

Insights drawn from EDA



Flight Number vs. Launch Site

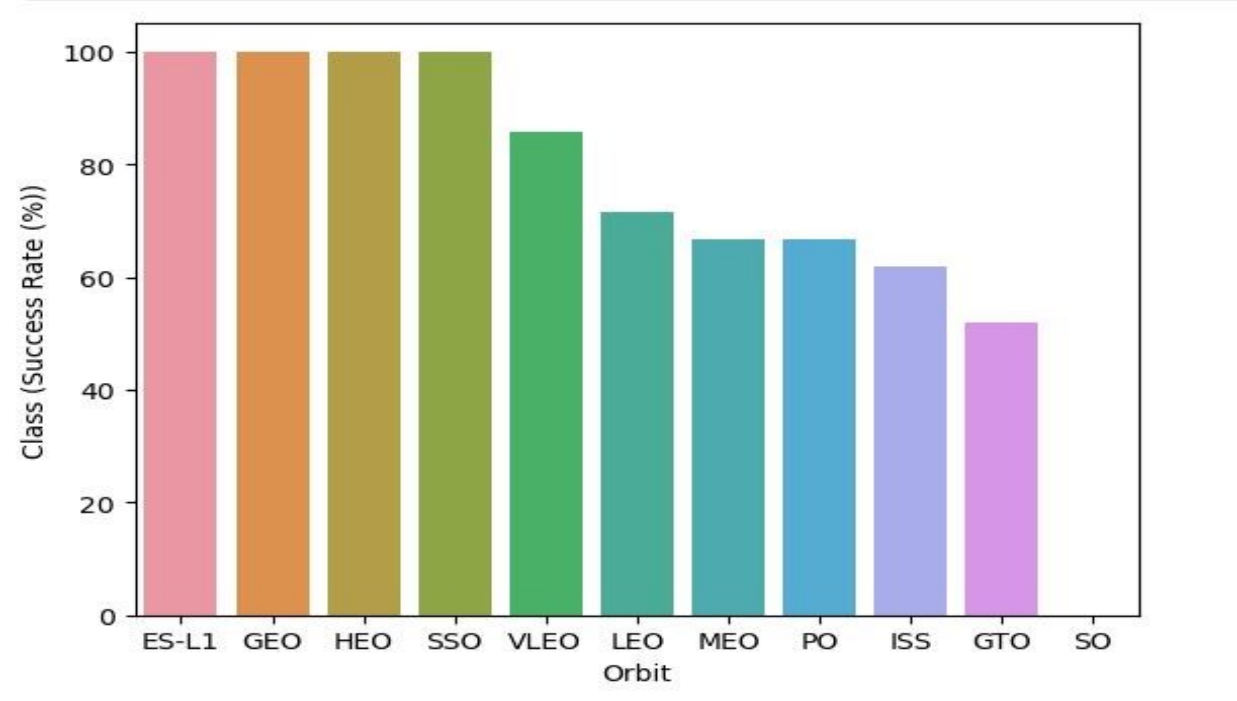


This scatter plot illustrates a correlation where a higher number of flights from a launch site corresponds to a greater success rate. Notably, site CCAFS SLC40 deviates from this pattern, showing the least discernible trend.

Payload vs. Launch Site

This scatter plot indicates that once the payload mass exceeds 7000 kg, the probability of success significantly increases. However, there is no evident pattern suggesting that the success rate is dependent on the launch site's relation to the payload mass.

Success Rate vs. Orbit Type

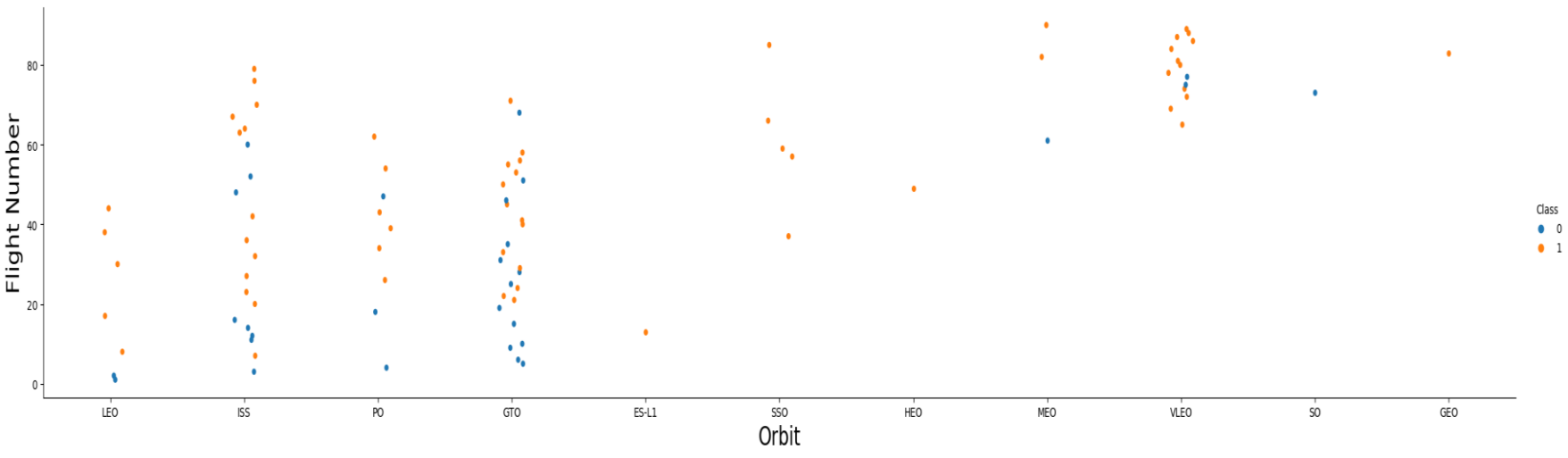


This figure illustrates the potential influence of orbits on landing outcomes, with certain orbits like SSO, HEO, GEO, and ES-L1 showing a 100% success rate, while SO orbit yields a 0% success rate.

However, a more in-depth analysis reveals that some of these orbits have only one occurrence, such as GEO, SO, HEO, and ES-L1.

This indicates that additional datasets are needed to discern any meaningful patterns or trends before drawing conclusive conclusions.

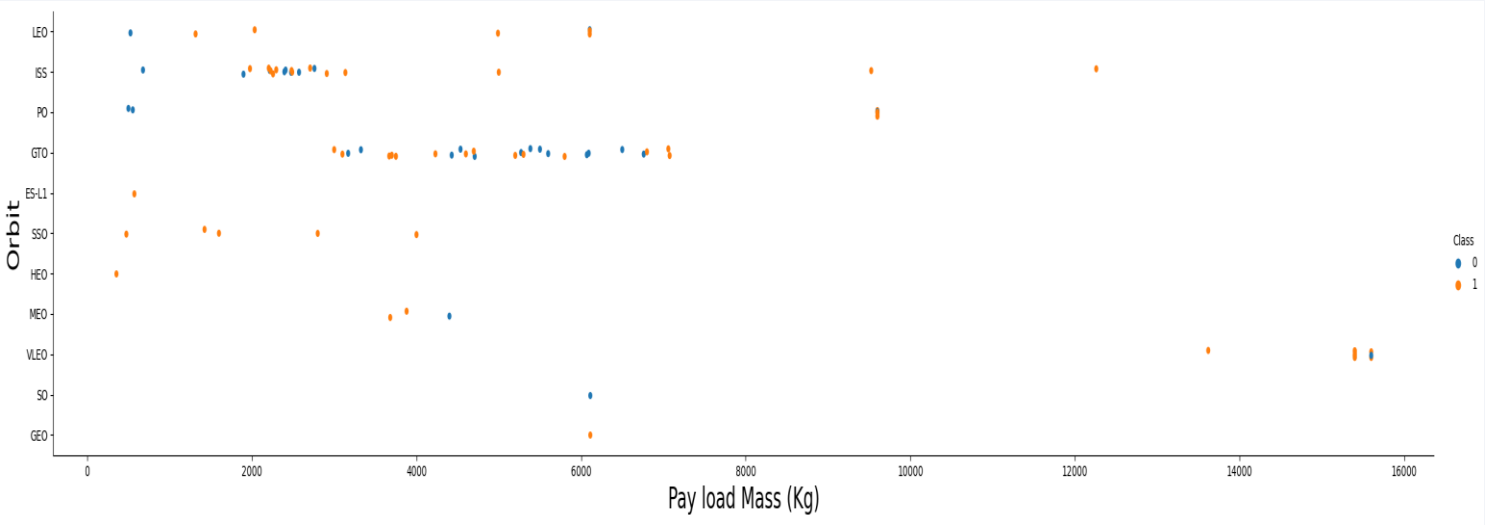
Flight Number vs. Orbit Type



This scatter plot demonstrates a general trend where a higher flight number on each orbit corresponds to a greater success rate, particularly in the LEO orbit.

However, the GTO orbit shows no discernible relationship between these attributes. It's important to note that orbits with only one occurrence should be excluded from the above statement, as they require additional datasets to draw meaningful conclusions.

Payload vs. Orbit Type

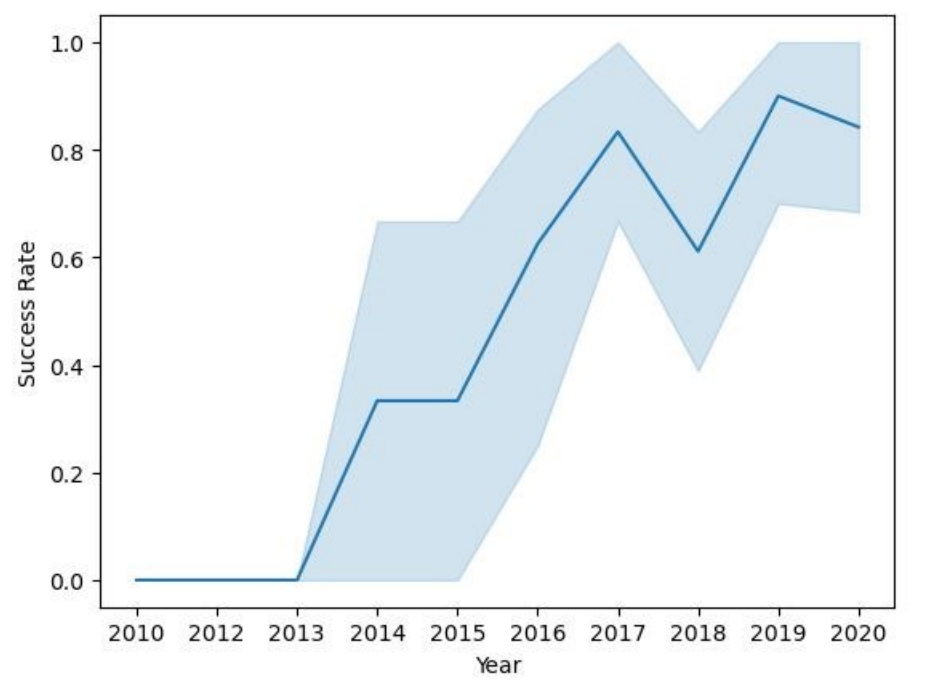


A heavier payload demonstrates a positive impact on LEO, ISS, and P0 orbits.

Conversely, it has a negative impact on MEO and VLEO orbits.

The GTO orbit appears to have no discernible relation between these attributes. Additionally, SO, GEO, and HEO orbits require more datasets to identify any potential patterns or trends.

Launch Success Yearly Trend



Increasing in trend from 2013 until 2020.

If this trend persists in the coming years, the success rate is projected to steadily increase, potentially reaching a 100% success rate.

All Launch Site Names

```
In [5]: %sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEX;
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

Out[5]:

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

The key **DISTINCT** is used to show the unique names of all the launch sites

Launch Site Names Begin with 'CCA'

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

In [11]:
```
task_2 = '''
        SELECT *
        FROM SpaceX
        WHERE LaunchSite LIKE 'CCA%'
        LIMIT 5
        '''
create_pandas_df(task_2, database=conn)
```

Out[11]:

	date	time	boosterversion	launchsite	payload	payloadmasskg	orbit	customer	missionoutcome	landingoutcome
0	2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
1	2010-08-12	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of...	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2	2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
3	2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
4	2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Query to display the 5 records with launch sites beginning with ‘CAA’

Total Payload Mass

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

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) AS "Total Payload Mass by NASA (CRS)
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3
sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

Total Payload Mass by NASA (CRS)
45596

Total payload carried by NASA is calculated as 45596

Average Payload Mass by F9 v1.1

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

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

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3
sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

Average Payload Mass by Booster Version F9 v1.1
2928

While the average payload carried by Falcon9 is 2928.4

First Successful Ground Landing Date

```
%sql SELECT MIN(DATE) AS "First Succesful Landing Outcome in Ground Pad
WHERE LANDING__OUTCOME = 'Success (ground pad)';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3
sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

First Succesful Landing Outcome in Ground Pad
2015-12-22

Min() function is used to find the result. The date of the first successful landing is on the 22nd of December 2015

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql SELECT BOOSTER_VERSION FROM SPACEX WHERE LANDING__OUTCOME = 'Success (drone ship)' \
AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000;
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

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

We utilized the WHERE clause to filter for boosters that have successfully landed on a drone ship. Additionally, we applied the AND condition to identify successful landings with a payload mass greater than 4000 but less than 6000.

Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes

```
%sql SELECT COUNT(MISSION_OUTCOME) AS "Successful Mission" FROM SPACEX WHERE MISSION_OUTCOME LIKE 'Success%';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

Successful Mission
100

```
%sql SELECT COUNT(MISSION_OUTCOME) AS "Failure Mission" FROM SPACEX WHERE MISSION_OUTCOME LIKE 'Failure%';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.clou
d:32731/bludb
Done.
```

Failure Mission
1

Wildcard like % is used to filter for **WHERE** the Mission_Outcome were successful

Boosters Carried Maximum Payload

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

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.clogj3sd0tgtu0lqde00.databases.appdomain.clou
d:32731/bludb
Done.
```

Booster Versions which carried the Maximum Payload Mass
F9 B5 B1048.4
F9 B5 B1048.5
F9 B5 B1049.4
F9 B5 B1049.5
F9 B5 B1049.7
F9 B5 B1051.3
F9 B5 B1051.4
F9 B5 B1051.6
F9 B5 B1056.4
F9 B5 B1058.3
F9 B5 B1060.2
F9 B5 B1060.3

The booster with the max payload is identified with **WHERE** and **MAX()** function

2015 Launch Records

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

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.
databases.appdomain.cloud:32731/bludb
Done.
```

booster_version	launch_site
F9 v1.1 B1012	CCAFS LC-40
F9 v1.1 B1015	CCAFS LC-40

WHERE, **LIKE**, **AND**, and **BETWEEN** used to filter for failed outcomes, their booster version and launch site names in 2015.

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

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

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

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

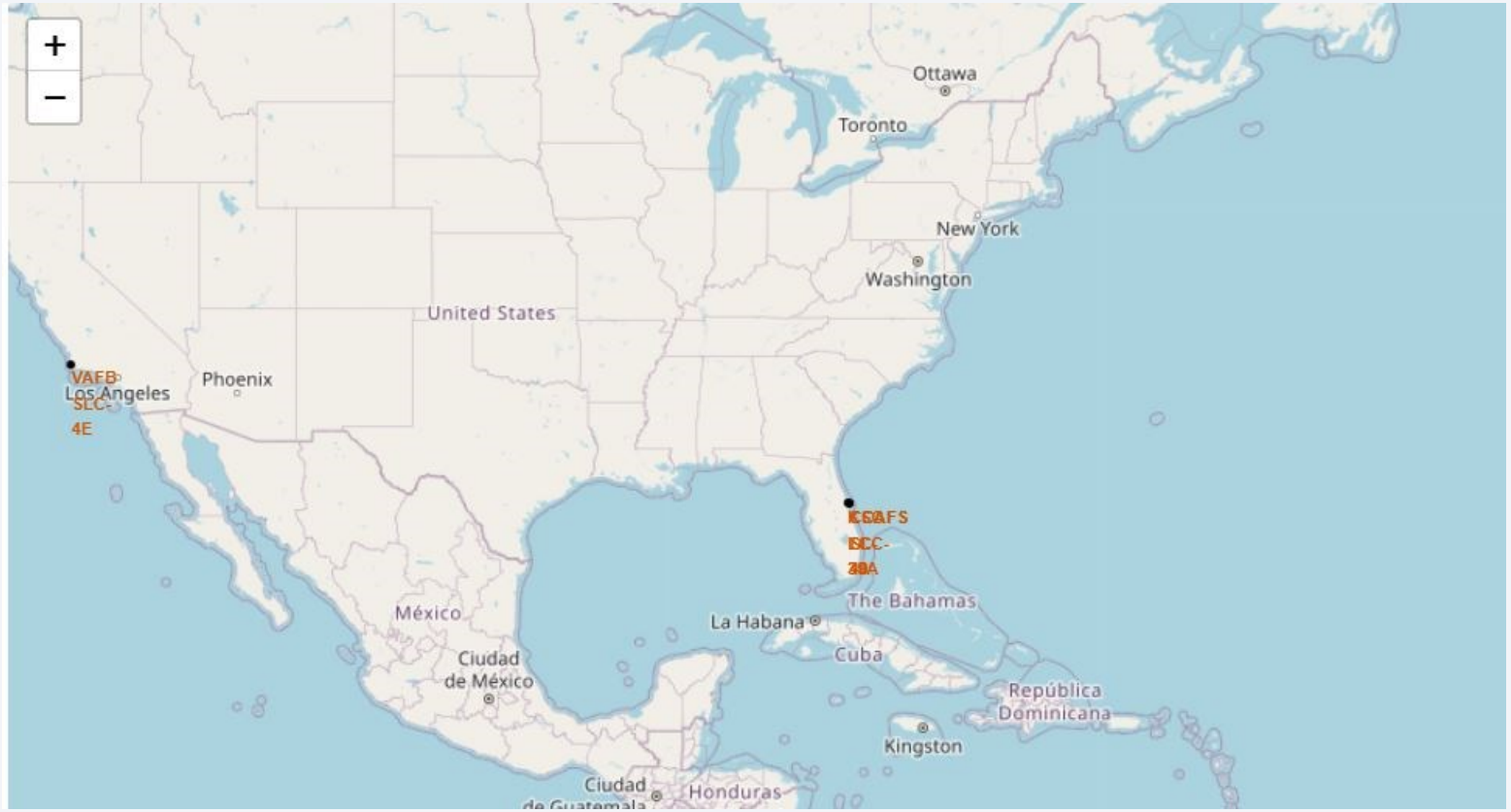
We chose Landing outcomes and the **COUNT** of landing outcomes from the data, using the **WHERE** clause to filter for landing outcomes between 2010-06-04 and 2010-03-20.

Subsequently, we applied the **GROUP BY** clause to group the landing outcomes and used the **ORDER BY** clause to arrange the grouped landing outcomes in descending order.

Section 3

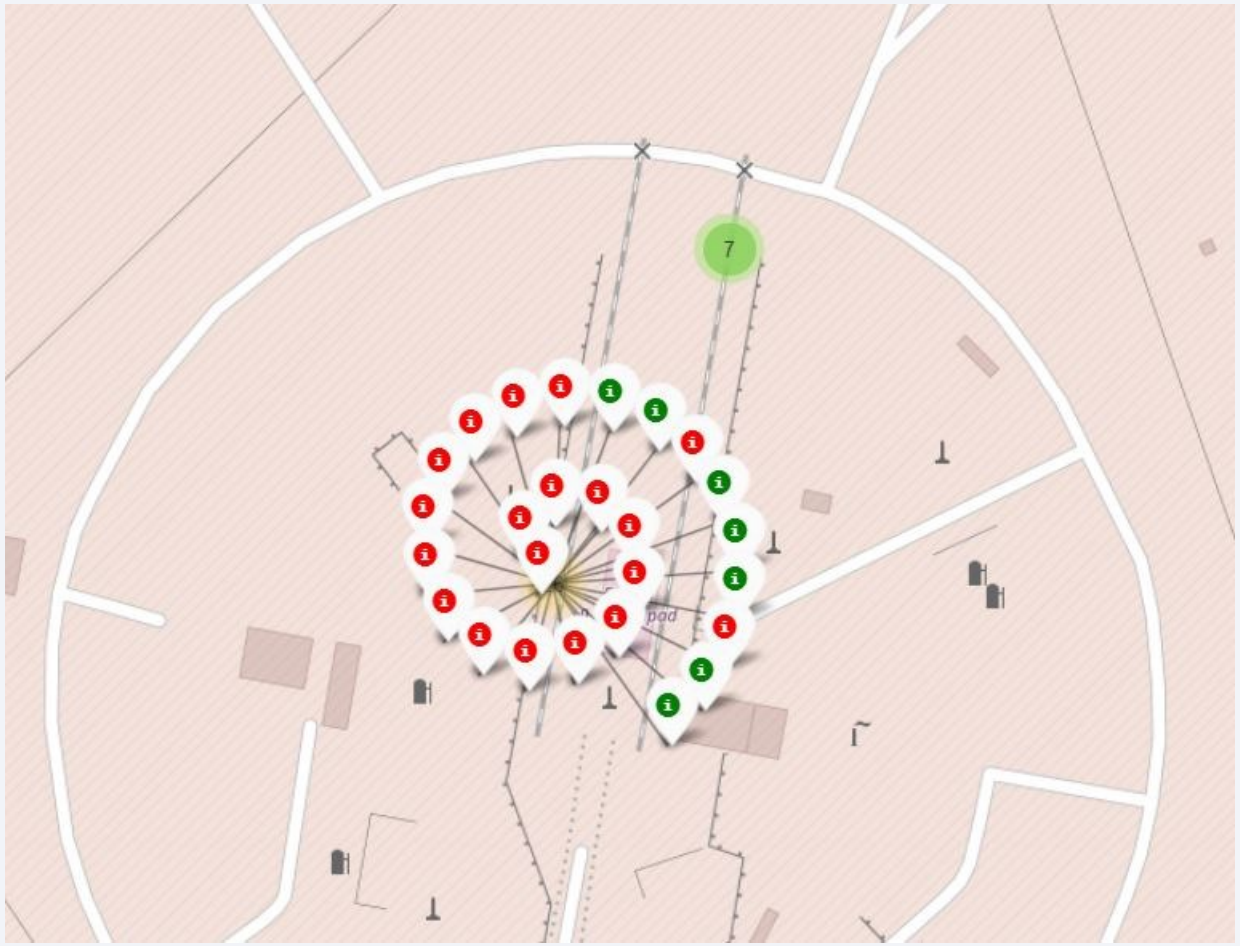
Launch Sites Proximities Analysis

Location of all the Launch Sites



SpaceX launch sites across the USA

Markers showing launch sites with color labels



Markers from the Florida Launch Site

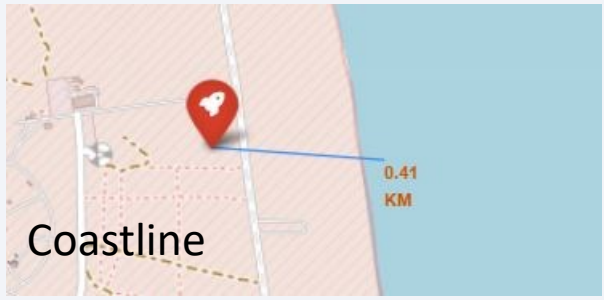
Green for successful launch site

Red for failure

Launch Sites Distance to Landmarks

Railway

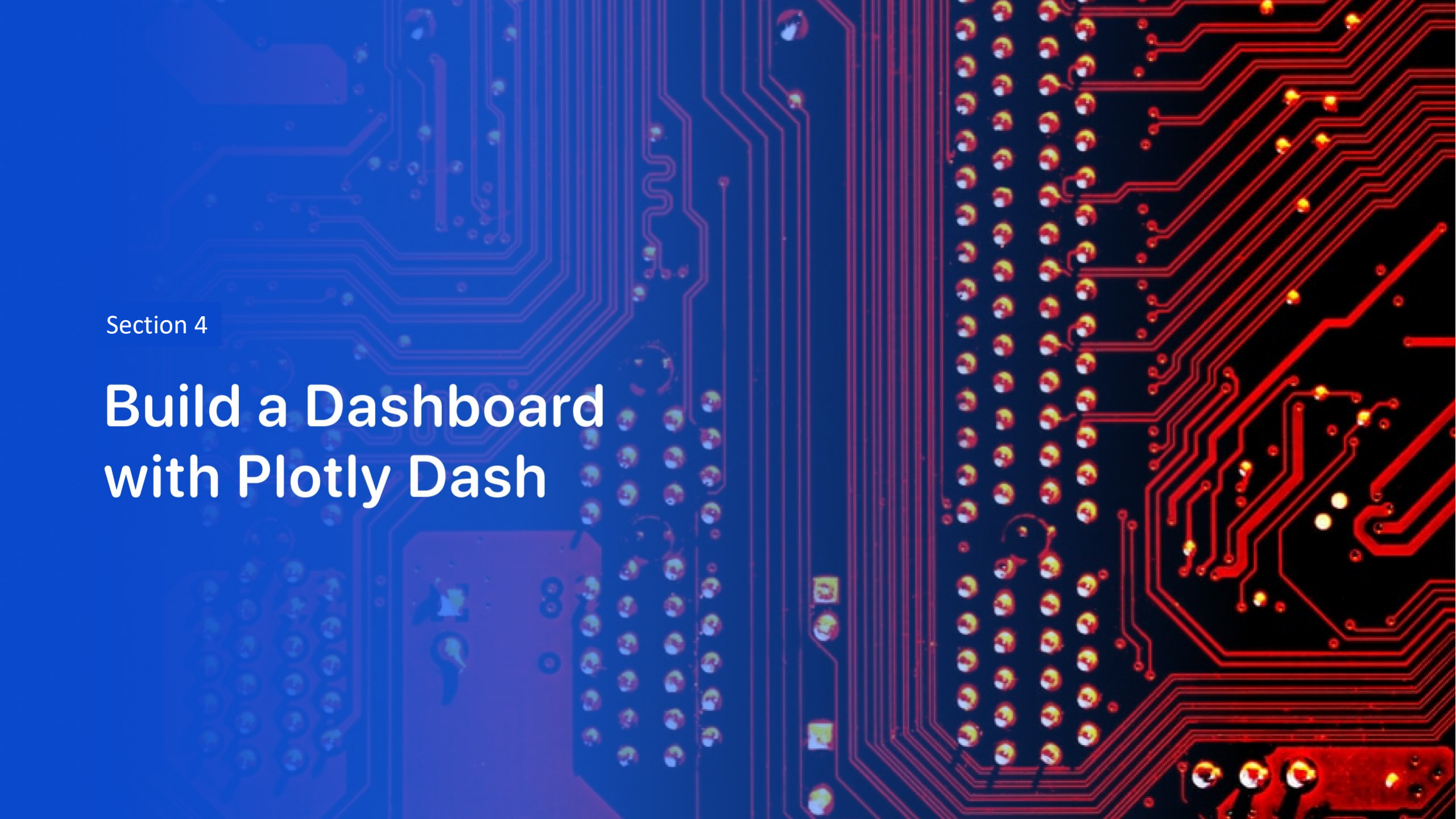
Parkway



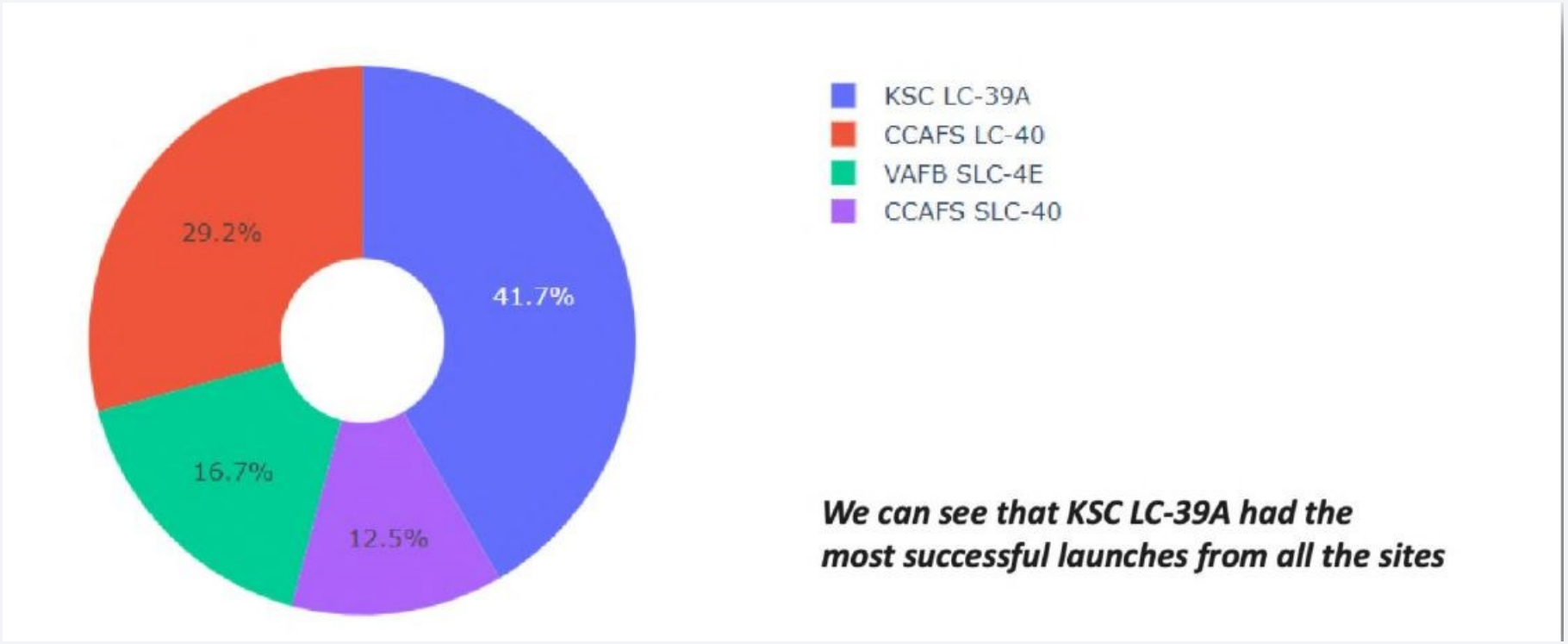
Coastline

City

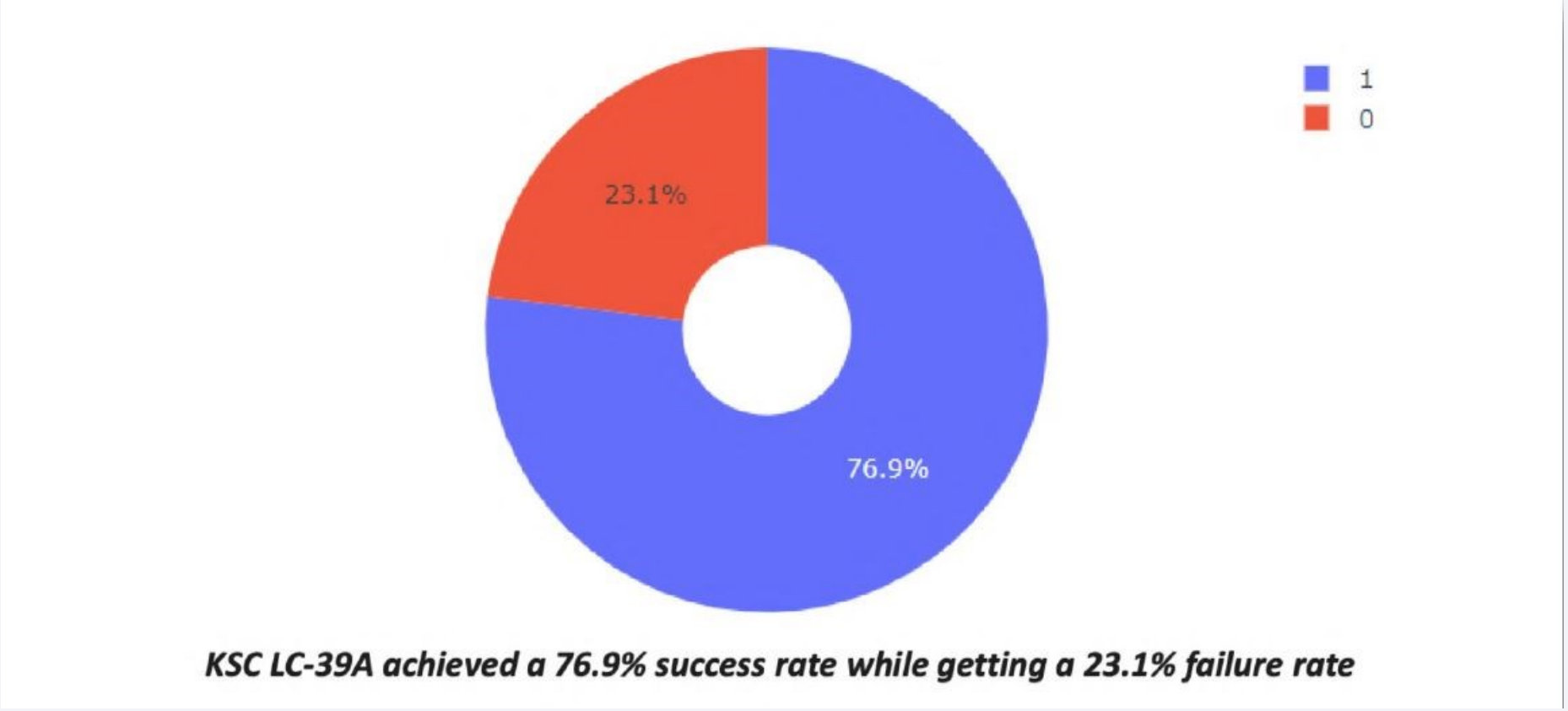
Launch sites are in close proximity to coastline


Section 4
Build a Dashboard with Plotly Dash

Percentage of success of each site

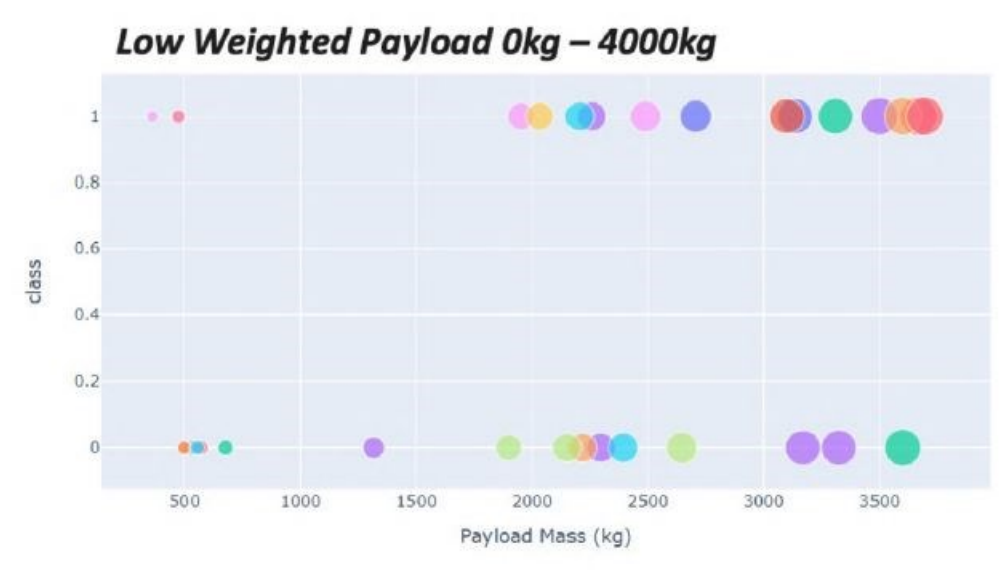


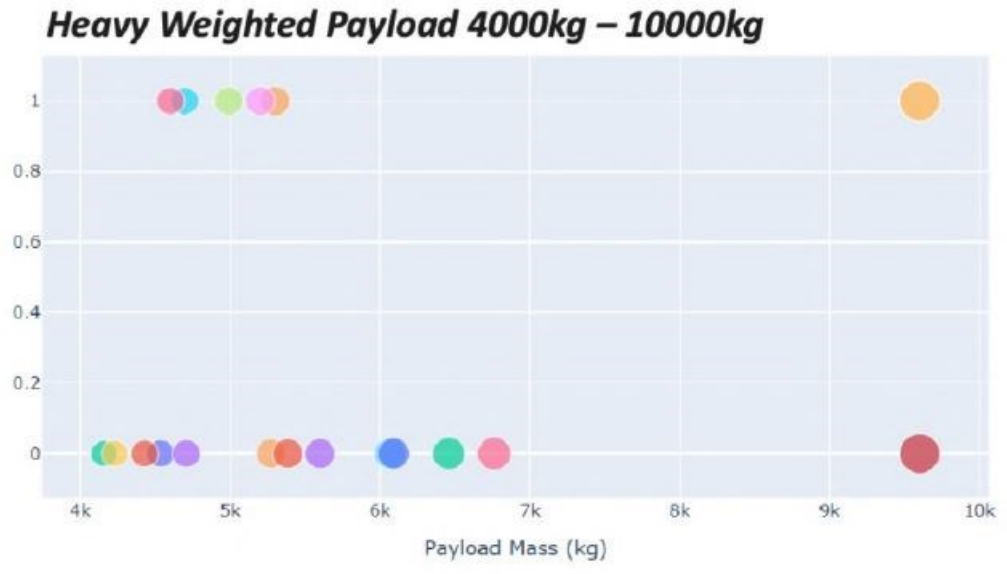
The highest launch-success ratio: KSC LC-39A



KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

Payload vs Launch Outcome Scatter Plot





all the success rate for low weighted payload is higher than heavy weighted payload

Section 5

Predictive Analysis (Classification)

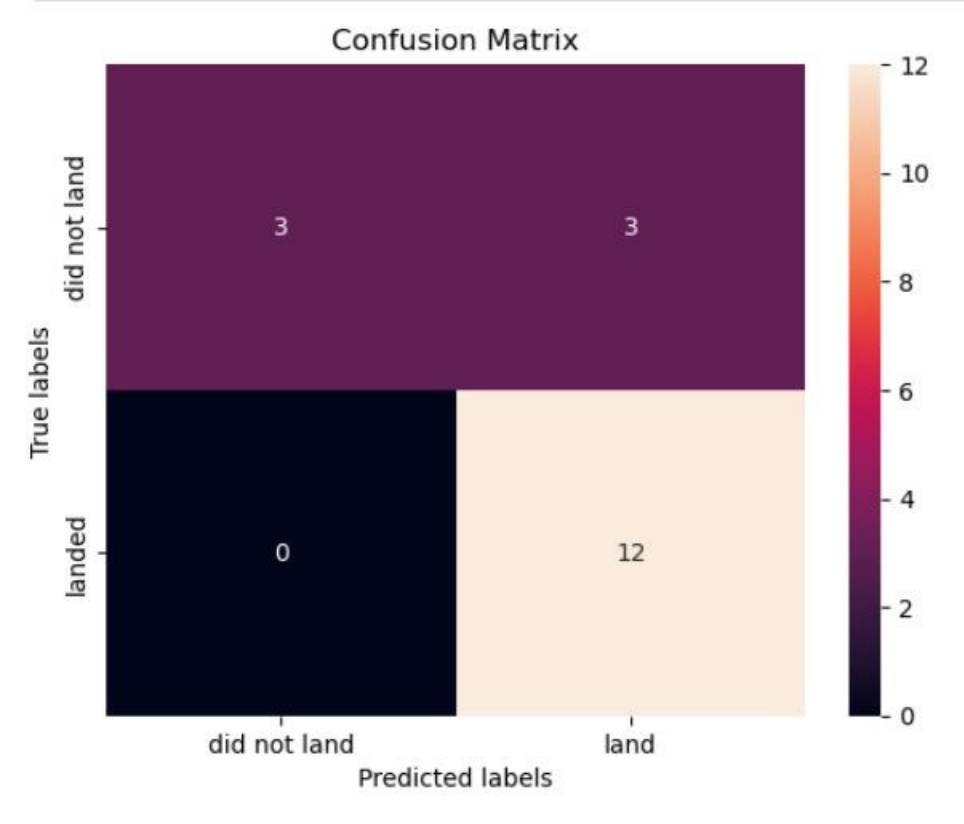
Classification Accuracy

```
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 0.9017857142857142
Best Params is : {'criterion': 'entropy', 'max_depth': 10, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_sampl
es_split': 10, 'splitter': 'random'}
```

The tree algorithm is the best one, as it has the highest classification accuracy

Confusion Matrix



The confusion matrix for the decision tree classifier shows that the classifier can distinguish between the different classes.

The major problem is the false positives, .i.e. unsuccessful landing marked as successful landing by the classifier.

Conclusions

Based on the analysis of the dataset, the following conclusions can be drawn:

•The Tree Classifier Algorithm emerges as the most effective machine learning approach for this dataset.

•Low-weighted payloads (defined as 4000kg and below) exhibit better performance than heavy-weighted payloads.

•Since 2013, the success rate for SpaceX launches has shown a consistent increase, suggesting a positive trend that is expected to enhance launches in the future.

•KSC LC-39A stands out with the highest success rate among all launch sites, at 76.9%.

•SSO orbit demonstrates the highest success rate, achieving 100%, and has more than one occurrence.

Thank you!