IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Raúl Acosta Murillo
18 Feb 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 in screenshots
  - Predictive Analytics result

Introduction

Project Background and Context

SpaceX has revolutionized the aerospace industry by significantly reducing the cost of rocket launches through the reuse of its first-stage boosters. Reusable rockets allow the company to save millions of dollars per launch, making space exploration more affordable and accessible. As SpaceX continues to dominate the market, other aerospace companies aiming to compete must consider similar cost-saving strategies. One critical factor in determining the cost of a launch is whether the first stage of the rocket successfully lands and can be reused.

This project aims to build a machine learning pipeline that predicts the likelihood of a successful first-stage landing. By leveraging historical launch data, the project seeks to identify key factors that influence landing outcomes, such as payload mass, launch site, booster version, and environmental conditions. Accurate predictions of landing success will help other companies estimate launch costs more precisely, enabling them to submit competitive bids against SpaceX for commercial satellite launches and other space missions.

Introduction

Problems You Want to Find Answers To

The project aims to answer several key questions related to SpaceX's rocket landings. Firstly, it seeks to identify the factors that most significantly influence the success of first-stage landings. This includes analysing features such as payload mass, launch site, booster version, orbit type, and weather conditions. Another essential question is whether machine learning can reliably predict the success of booster landings based on historical data.

Additionally, the project explores how different launch conditions, including various launch sites, payload sizes, and booster versions, affect the likelihood of a successful landing. Interactive visualizations, created using tools like Folium, will also be used to provide a deeper understanding of SpaceX's launch patterns and outcomes through dynamic maps and charts. Finally, the project evaluates the accuracy of the machine learning models used and seeks ways to refine them for better performance in real-world applications. By addressing these challenges, the project offers valuable insights into SpaceX's operations and provides a competitive edge to other aerospace companies aiming to enter the commercial launch market.

Section 1

Methodology

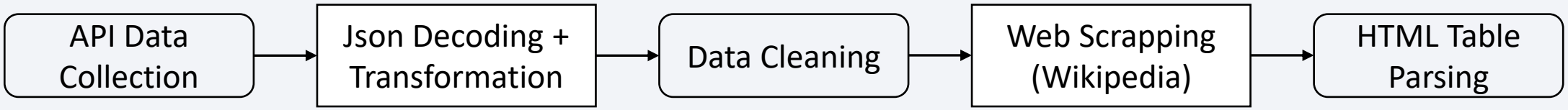
Methodology

Executive Summary

- Data collection methodology:
  - Data Collection through API and Web Scraping
  - Perform data wrangling
  - 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

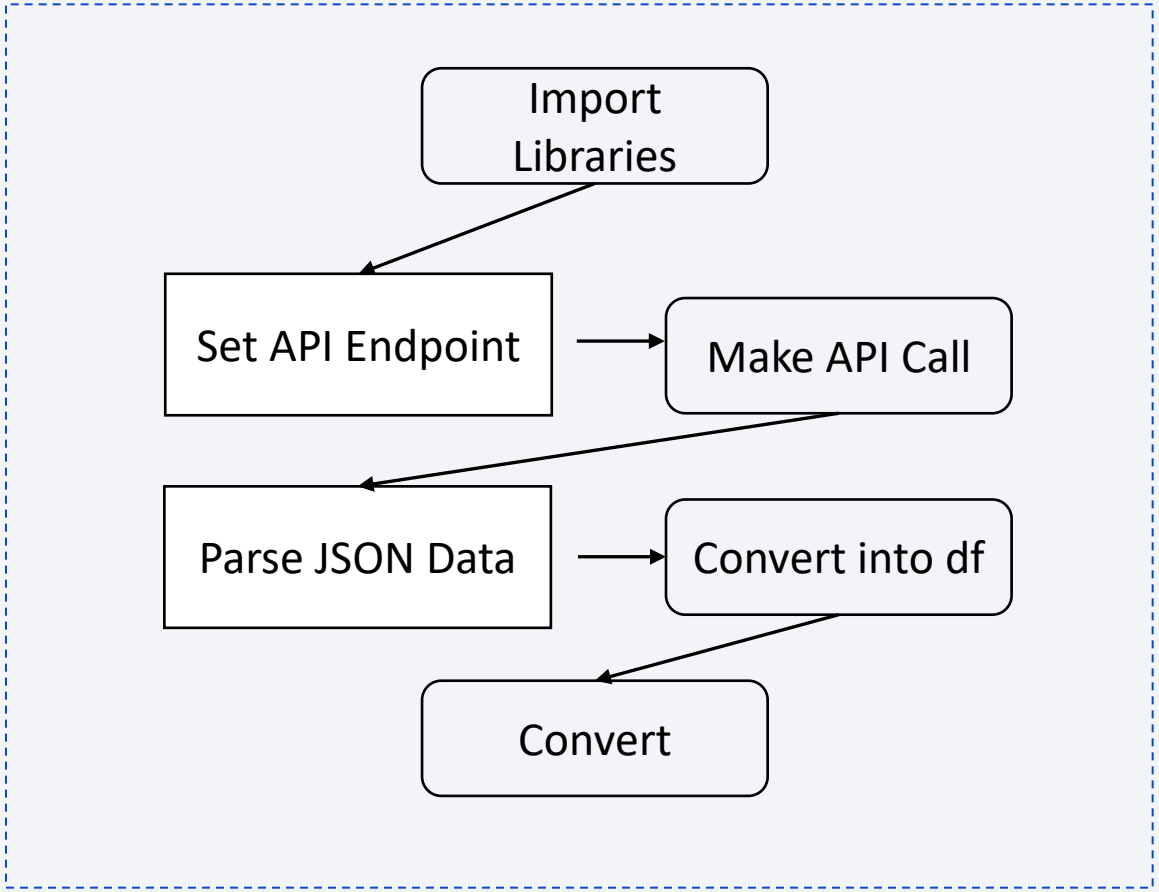
- The data for this project was collected using multiple methods to ensure comprehensive and accurate information. Initially, data collection was performed through a GET request to the SpaceX API, allowing access to detailed records of Falcon 9 launches. The API response, returned in JSON format, was decoded using the `.json()` function and then transformed into a structured pandas dataframe with the help of the `.json_normalize()` function.
- Once the data was gathered, it was carefully cleaned by checking for missing values and filling them in where necessary to maintain data integrity. Additionally, web scraping techniques were employed to extract Falcon 9 launch records from Wikipedia. Using the BeautifulSoup library, the launch records were obtained from an HTML table, parsed, and subsequently converted into a pandas dataframe. This multi-faceted approach ensured that the dataset was well-prepared for further analysis and machine learning tasks.



Data Collection – SpaceX API

- We utilized a GET request to retrieve data from the SpaceX API, followed by cleaning the obtained data and performing essential data wrangling and formatting.

- https://github.com/RollerCoaster1899/IBM_Applied_Data_Science/blob/main/1_Data-Collection_SpaceX-API.ipynb



Data Collection – SpaceX API

Task 1: Request and parse the SpaceX launch data using the GET request

To make the requested JSON results more consistent, we will use the following static response object for this project:

```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_api.json'
```

[9] ✓ 0.0s Python

We should see that the request was successfull with the 200 status response code

```
response.status_code
```

[10] ✓ 0.0s Python

```
200
```

Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

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

[11] ✓ 0.0s Python

Using the dataframe `data` print the first 5 rows

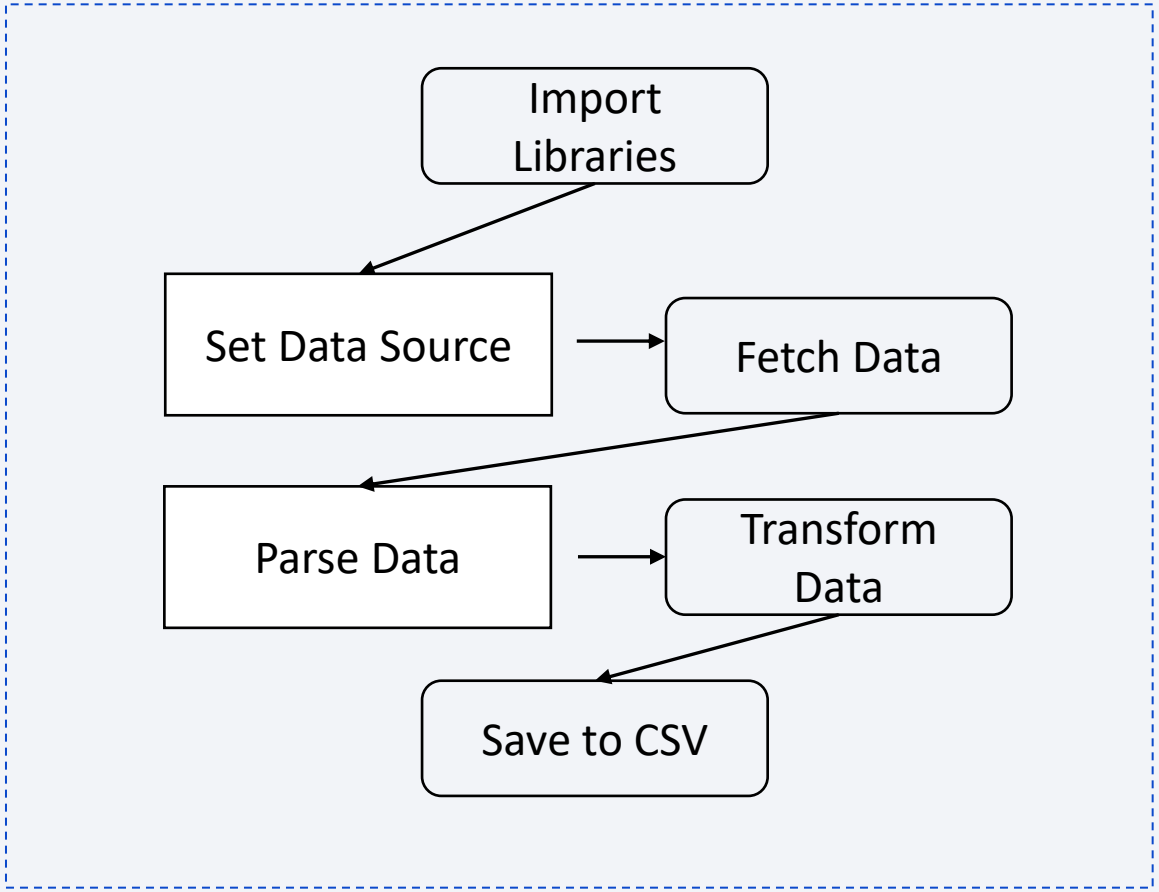
```
# Get the head of the dataframe
data.head()
```

[12] ✓ 0.0s Python

static_fire_date_utc	static_fire_date_unix	net	window	rocket	success	failures	details	crew	ships	capsules	payloads
						[{'time': 33, 'altitude':	Engine				

Data Collection - Scraping

- We used web scraping with BeautifulSoup to extract Falcon 9 launch records from a Wikipedia. The extracted table was then parsed and converted into a pandas df.
- https://github.com/RollerCoaster1899/IBM_Applied_Data_Science/blob/main/2_Data-Collection_Web-Scrapping.ipynb



Data Collection - Scraping

TASK 1: Request the Falcon9 Launch Wiki page from its URL

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

```
[5]
# use requests.get() method with the provided static_url
# assign the response to a object
data  = requests.get(static_url).text
```

Create a `BeautifulSoup` object from the HTML `response`

```
[6]
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(data, 'html5lib')
```

Print the page title to verify if the `BeautifulSoup` object was created properly

```
[7]
# Use soup.title attribute
print(soup.title)
```

```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

Data Collection - Scraping

TASK 2: Extract all column/variable names from the HTML table header

Next, we want to collect all relevant column names from the HTML table header

Let's try to find all tables on the wiki page first. If you need to refresh your memory about `BeautifulSoup`, please check the external reference link towards the end of this lab

[8]
```
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = soup.find_all('table')
```

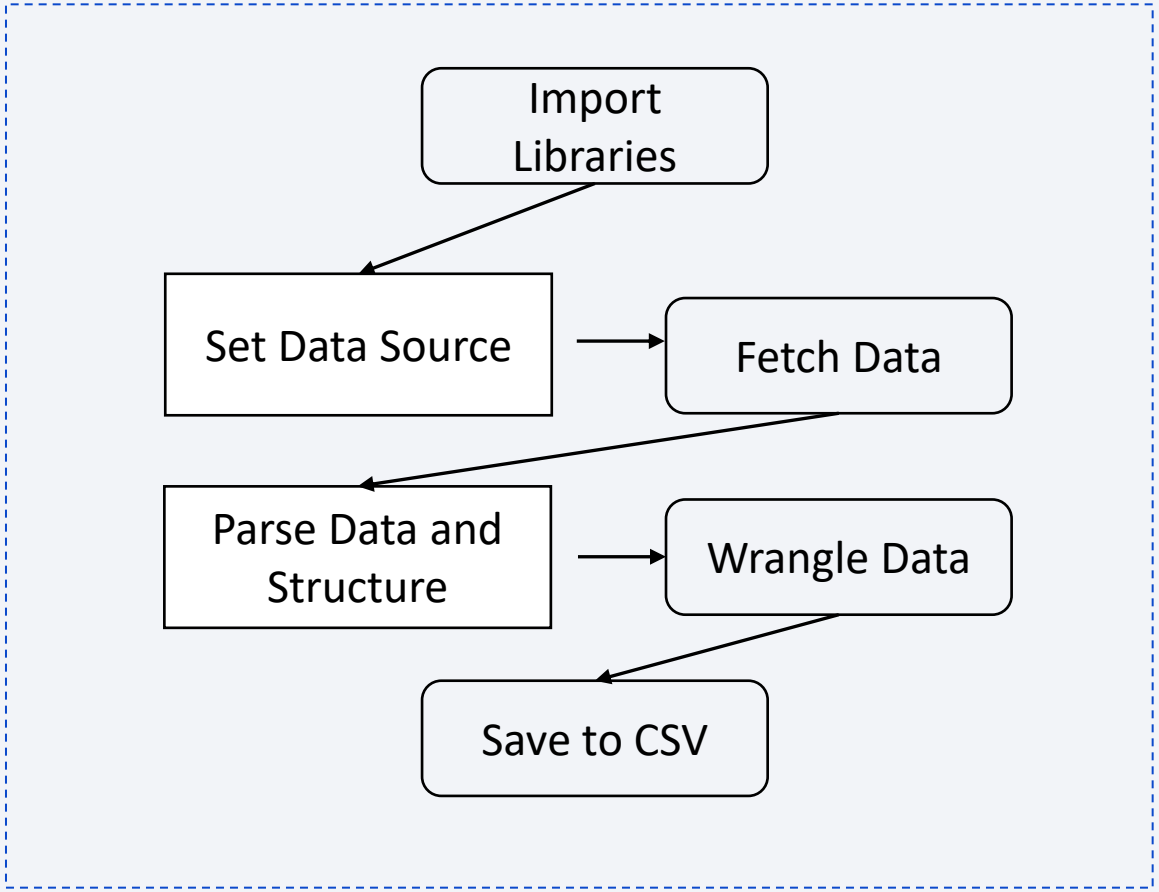
Starting from the third table is our target table contains the actual launch records.

[9]
```
# Let's print the third table and check its content
first_launch_table = html_tables[2]
print(first_launch_table)
```

```
<table class="wikitable plainrowheaders collapsible" style="width: 100%;">
<tbody><tr>
<th scope="col">Flight No.
</th>
<th scope="col">Date and<br/>time (<a href="/wiki/Coordinated_Universal_Time" title="Coordinated Universal Time">UTC</a>)
</th>
<th scope="col"><a href="/wiki/List_of_Falcon_9_first-stage_boosters" title="List of Falcon 9 first-stage boosters">Version,<br/>Booster</a>
</th>
<th scope="col">Launch site
```

Data Wrangling

The project methodology involved collecting Falcon 9 launch data through API calls using requests and web scraping with BeautifulSoup, followed by data cleaning, preprocessing, and merging to create a ready-to-use dataset for analysis.



https://github.com/RollerCoaster1899/IBM_Applied_Data_Science/blob/main/3_%20Data%20Wrangling.ipynb

Data Wrangling

Data Analysis

Load Space X dataset, from last section.

```
df=pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/dataset_part_1.csv")
df.head(10)
```

[2] ✓ 0.4s

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003	-80.577366
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005	-80.577366

Identify and calculate the percentage of the missing values in each attribute

```
df.isnull().sum()/df.count()*100
```

[3] ✓ 0.0s

```
FlightNumber      0.000
Date              0.000
BoosterVersion    0.000
PayloadMass       0.000
Orbit             0.000
LaunchSite        0.000
Outcome           0.000
Flights           0.000
GridFins          0.000
Reused            0.000
Legs              0.000
LandingPad       40.625
Block             0.000
ReusedCount       0.000
Serial            0.000
Longitude         0.000
Latitude          0.000
dtype: float64
```

Identify which columns are numerical and categorical:

```
df.dtypes
```

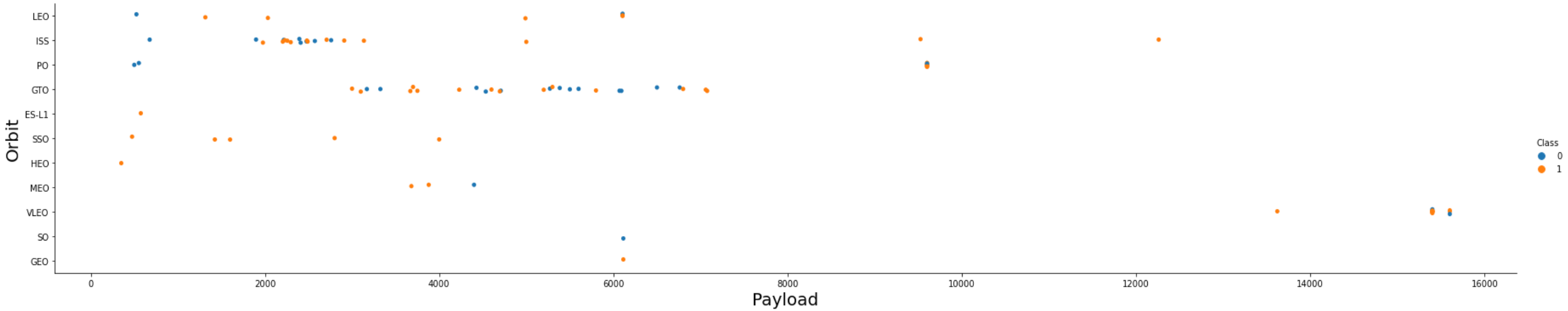
[4] ✓ 0.0s

```
FlightNumber      int64
Date             object
BoosterVersion   object
PayloadMass     float64
Orbit            object
LaunchSite       object
Outcome          object
Flights           int64
GridFins           bool
Reused             bool
Legs               bool
LandingPad       object
Block           float64
ReusedCount       int64
Serial           object
Longitude       float64
Latitude        float64
dtype: object
```

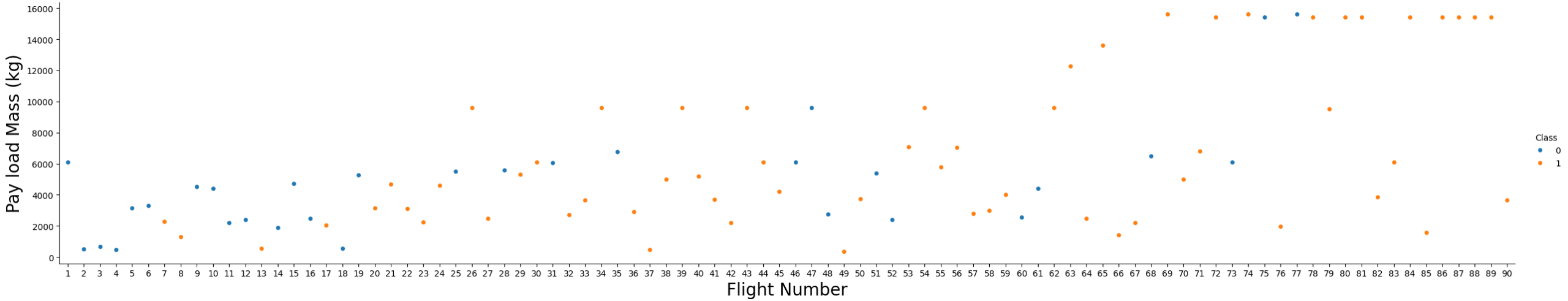
EDA with Data Visualization

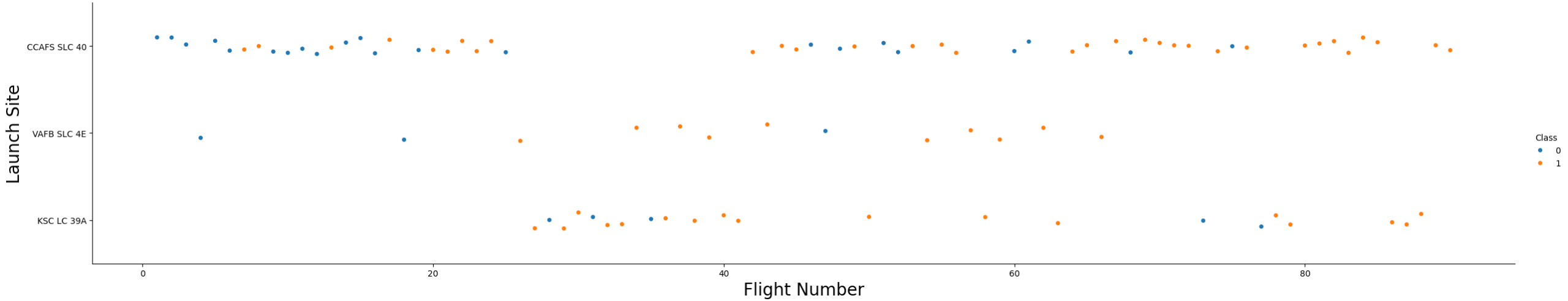
- The EDA methodology included the use of line plots to analyse trends over time, such as launch success rates across different periods, and categorical plots to compare key categories like launch sites and their associated outcomes. These charts were chosen to provide clear visualizations of the distribution, trends, and relationships within the Falcon 9 launch data, making it easier to interpret and derive insights.

https://github.com/RollerCoaster1899/IBM_Applied_Data_Science/blob/main/4_EDA_Data-Visualization.ipynb

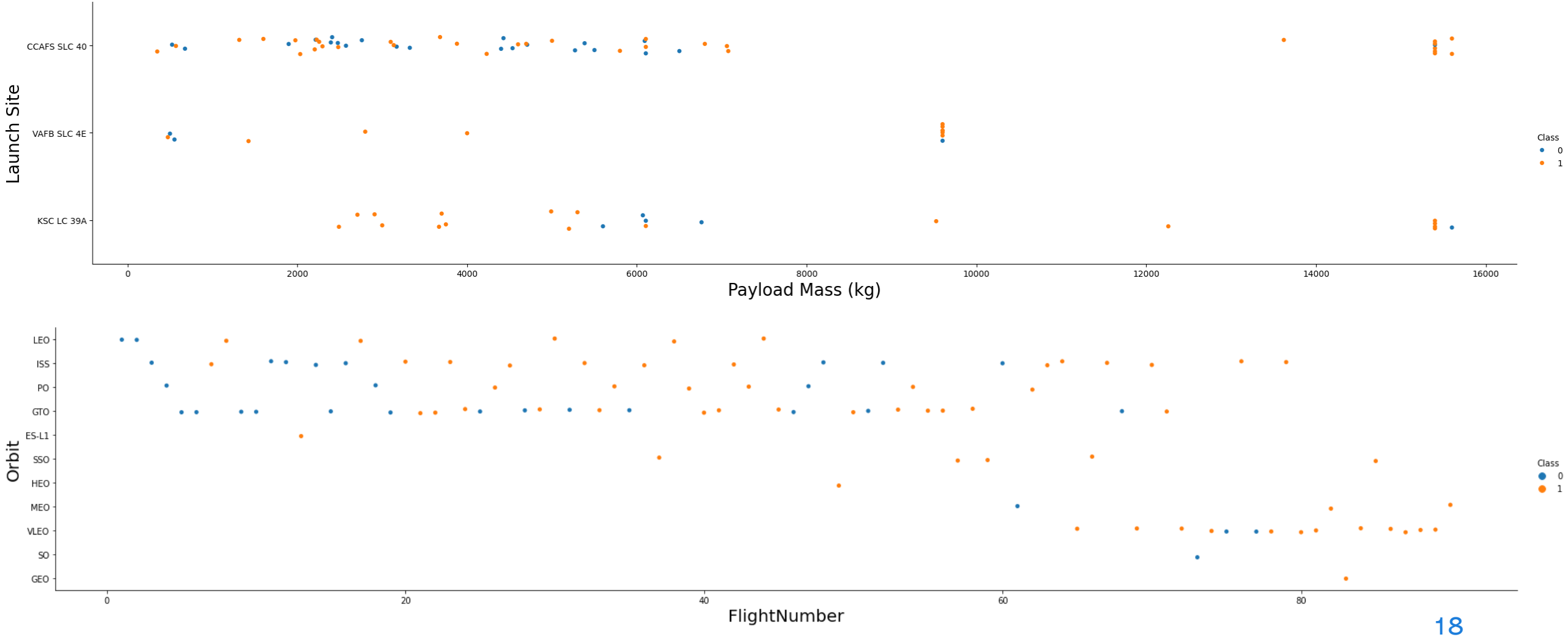


EDA with Data Visualization





EDA with Data Visualization



EDA with SQL

Here's a summary of the SQL queries performed:

- Performed counts to summarize the number of launches and successes.
- Retrieved maximum and minimum payload masses for analysis.
- Calculated average payload mass for different launch sites.
- Filtered data based on specific conditions like launch site or success status.
- Grouped data by launch sites and outcomes for aggregated analysis.

https://github.com/RollerCoaster1899/IBM_Applied_Data_Science/blob/main/5_EDA_SQL.ipynb

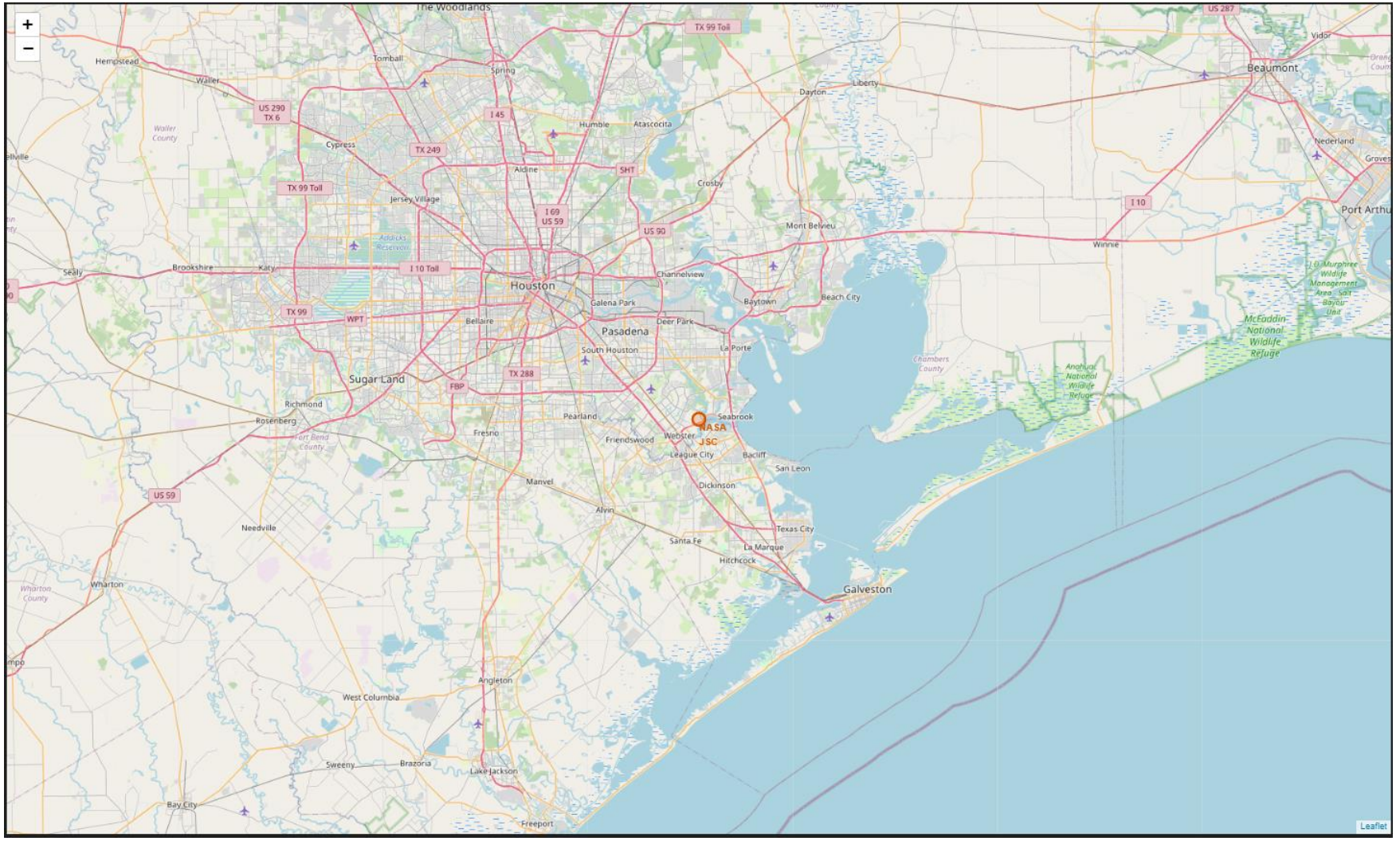
Build an Interactive Map with Folium

- In the Folium map created for this project, several map objects such as markers, circles, and lines were added to enhance the visualization of SpaceX launch data. Markers were placed at key launch sites to highlight the specific locations from which Falcon 9 rockets were launched. Circles were used to represent the radius around landing zones, providing a visual indication of proximity and landing accuracy. Lines were added to show the flight paths from launch sites to landing pads, helping to illustrate the trajectory and distance covered by each launch.
- These objects were added to the Folium map to provide an interactive and intuitive way to explore SpaceX's launch operations. Markers allowed users to easily identify and access information about each launch site, circles emphasized the landing areas and success rates, and lines depicted the rocket’s journey from liftoff to landing. This enriched visualization made it easier to analyse and understand the spatial relationships and operational scope of SpaceX's missions.

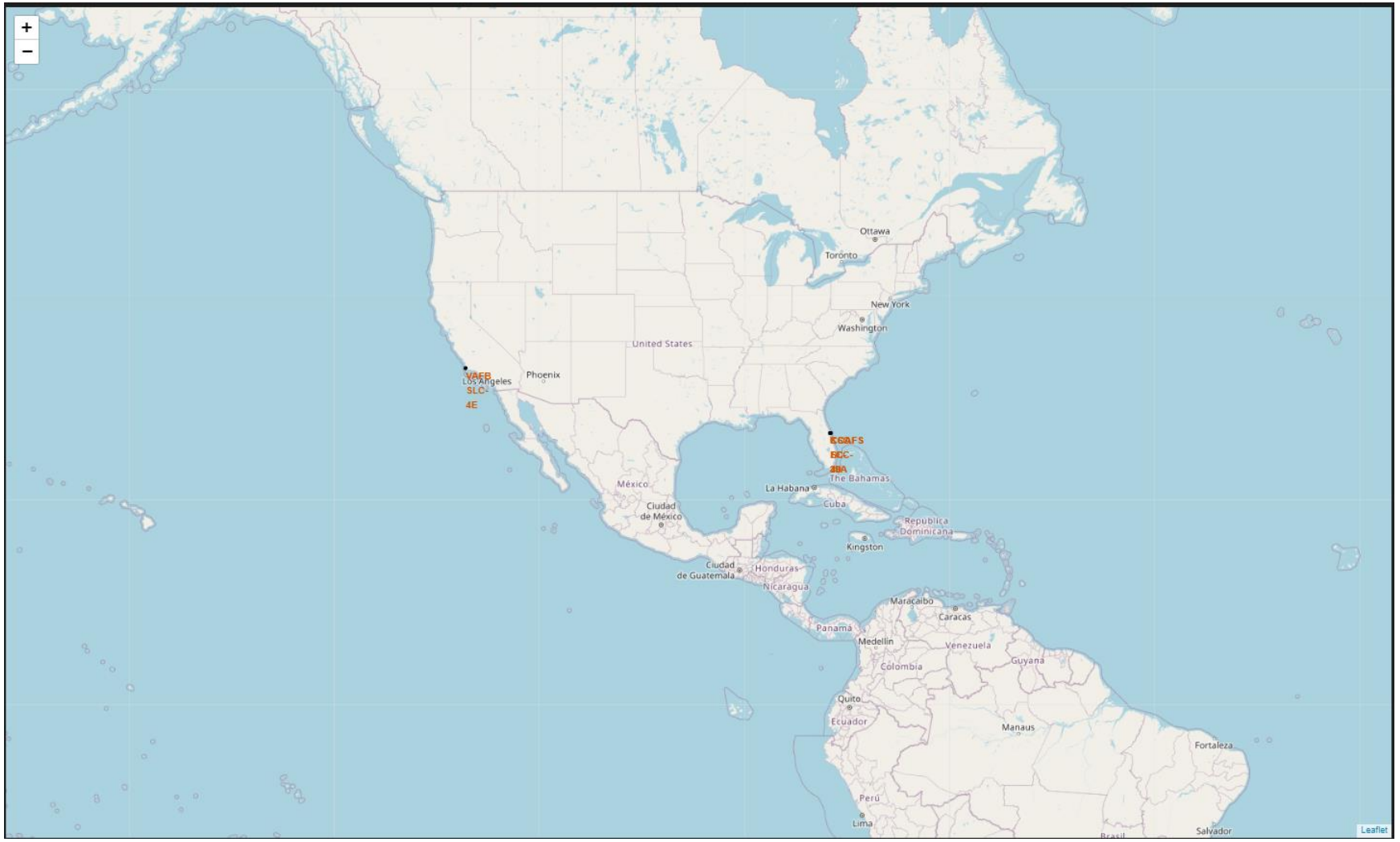
Build an Interactive Map with Folium

- https://github.com/RollerCoaster1899/IBM_Applied_Data_Science/blob/main/6_Folium.ipynb
- Add the GitHub URL of your completed interactive map with Folium map, as an external reference and peer-review purpose

Build an Interactive Map with Folium



Build an Interactive Map with Folium



Build a Dashboard with Plotly Dash

- An interactive dashboard was developed using Plotly Dash. Pie charts were created to display the total number of launches from various launch sites, providing a clear overview of launch distribution. Additionally, scatter plots were generated to illustrate the relationship between launch outcomes and payload mass (in kilograms) across different booster versions, offering insights into how payload weight affects mission success for each booster type.

- https://github.com/RollerCoaster1899/IBM_Applied_Data_Science/blob/main/7_Dashoard_Plotly-Dash.ipynb

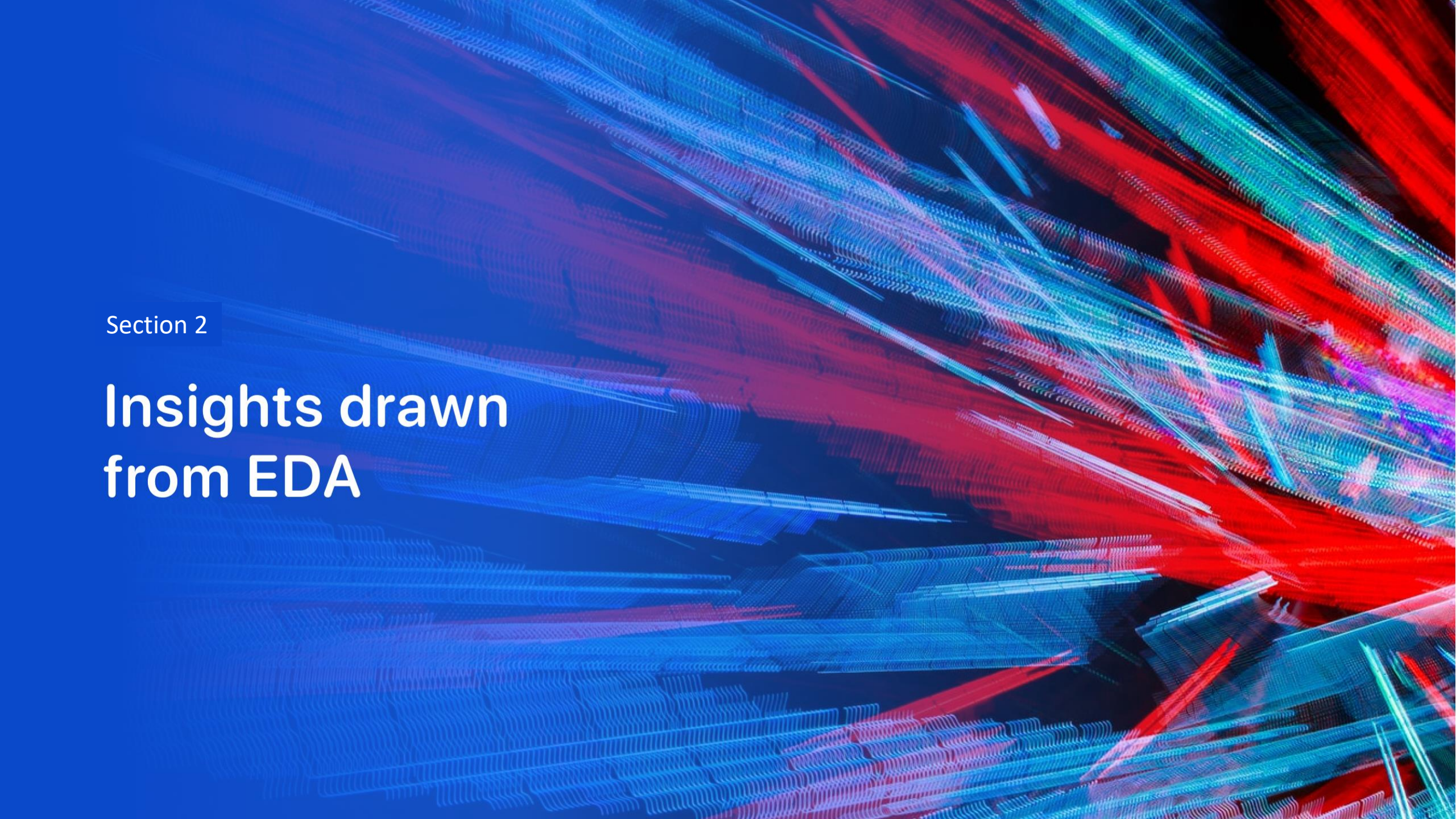
Predictive Analysis (Classification)

- The classification model development process began with data preparation, where the dataset was cleaned, preprocessed, and split into training and testing sets to ensure reliable evaluation. Multiple classification algorithms, including Logistic Regression, Decision Trees, Random Forest, and Support Vector Machines, were then selected and tested. Each model's performance was evaluated using key metrics such as accuracy, precision, recall, and F1-score to determine its effectiveness. To enhance the models further, hyperparameter tuning was performed using grid search and cross-validation techniques, which optimized the model parameters for better performance. Finally, the model with the highest evaluation scores was selected as the best-performing classification model, providing accurate predictions for SpaceX's first-stage booster landings.

- https://github.com/RollerCoaster1899/IBM_Applied_Data_Science/blob/main/8_Machine-Learning_Prediction.ipynb

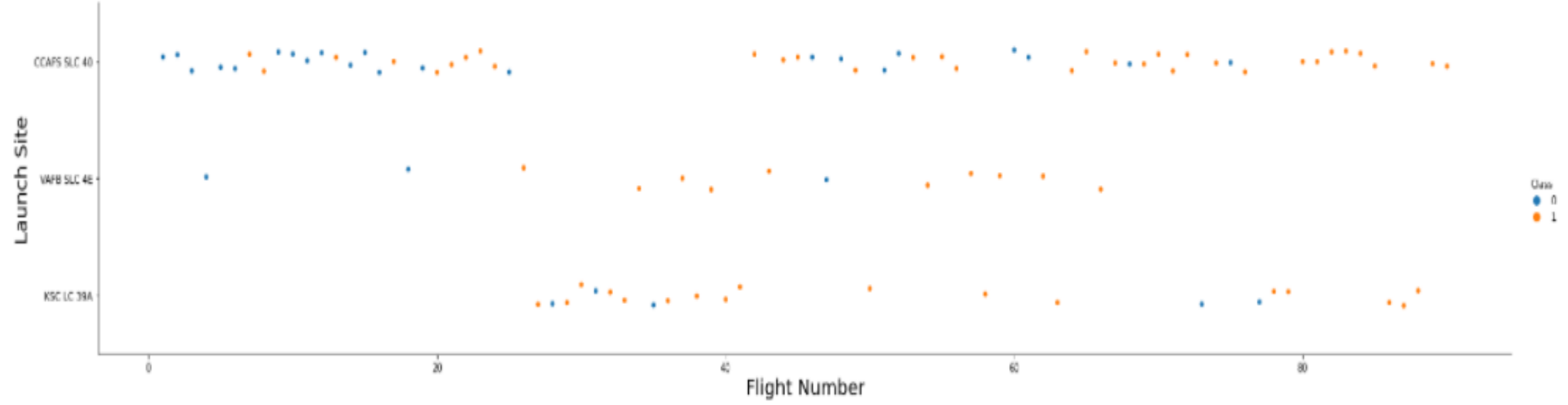
Results

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


Section 2
Insights drawn from EDA

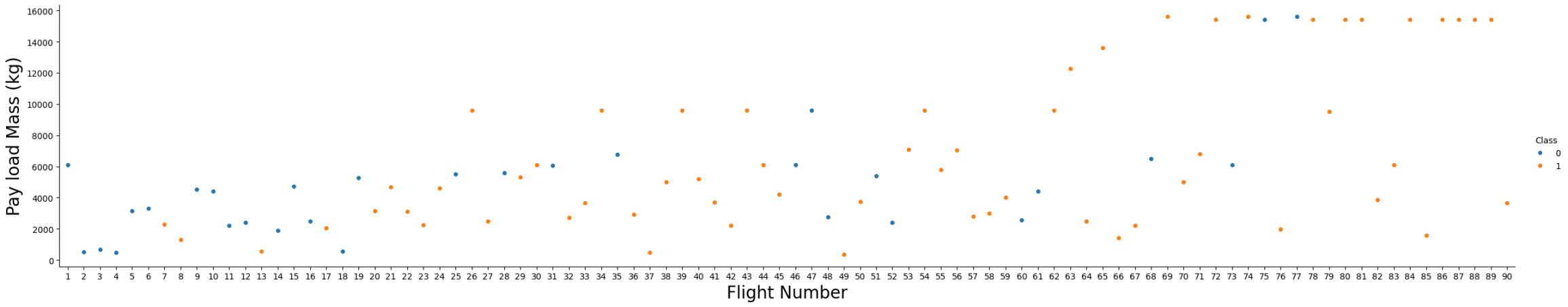
Flight Number vs. Launch Site

- The scatter plot shows that CCAFS SLC 40 had the highest number of launches, with many successful landings. KSC LC 39A also had a notable success rate, while VAFB SLC 4E had fewer launches with mixed outcomes. The trend indicates that sites with more launches tend to have higher success rates, likely due to experience and operational improvements.



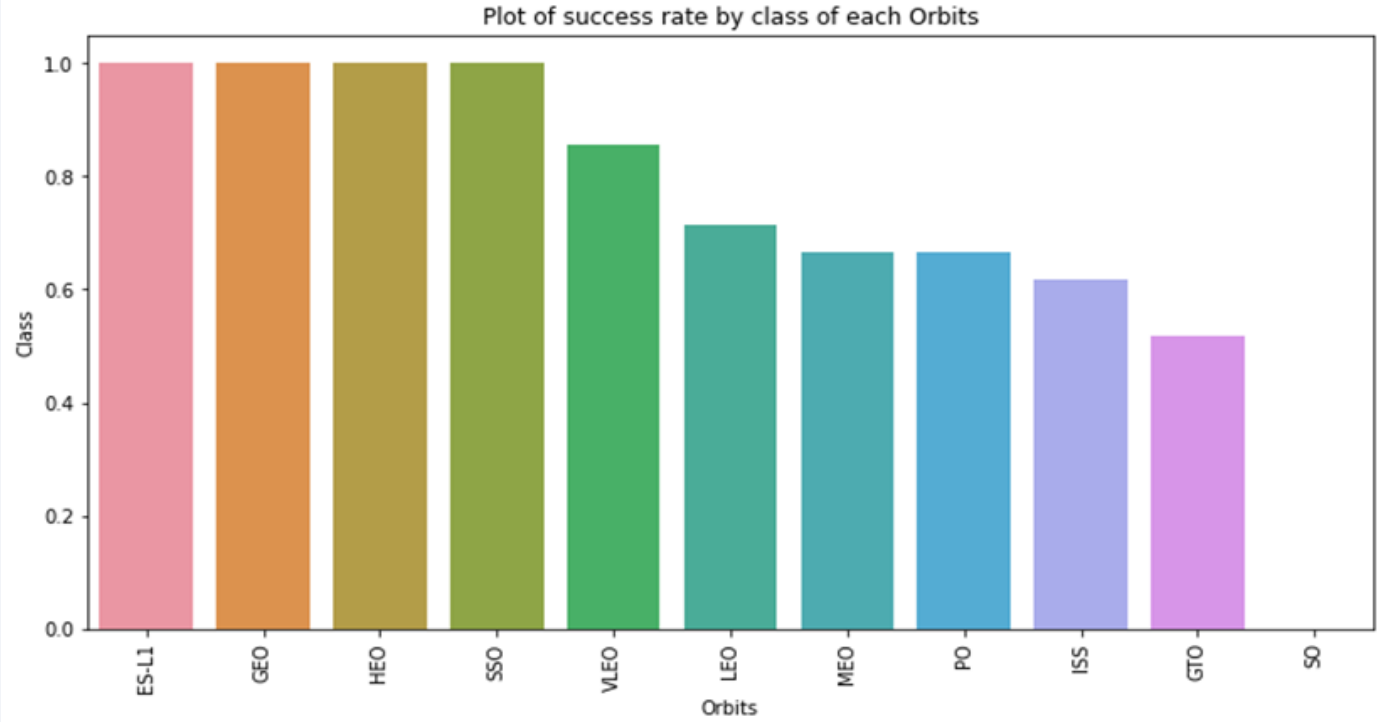
Payload vs. Launch Site

- The scatter plot shows the relationship between flight numbers and launch sites, with successful landings marked in orange and unsuccessful ones in blue. CCAFS SLC 40 had the highest number of launches and a growing trend of successful landings over time. KSC LC 39A also showed improved success rates with more launches, while VAFB SLC 4E had fewer launches with mixed results. The plot indicates that higher launch frequency is associated with increased landing success, reflecting continuous operational improvements and learning over time.



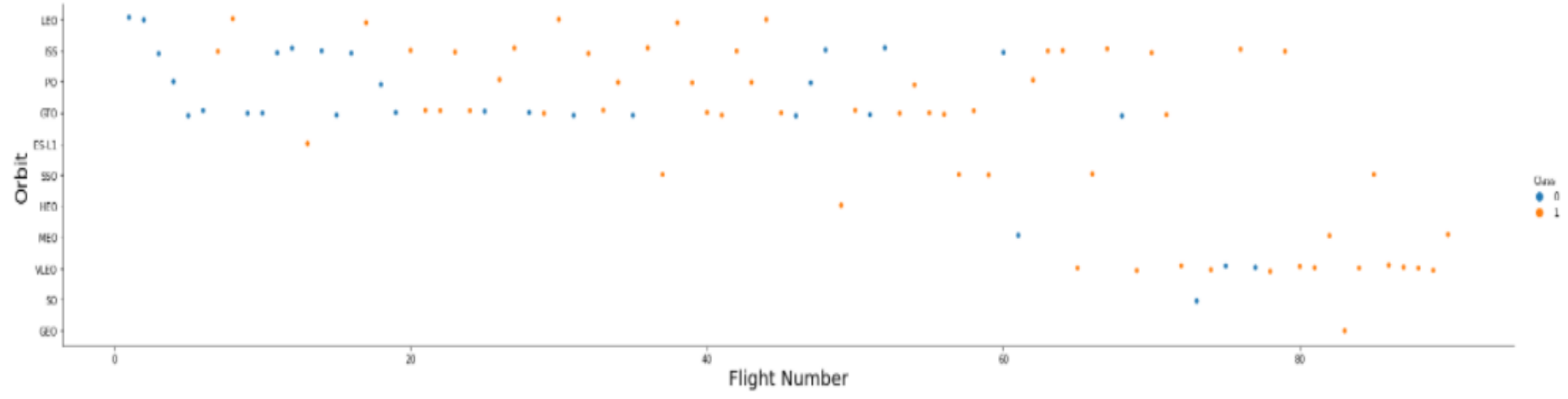
Success Rate vs. Orbit Type

- The bar plot shows that SpaceX rocket launches to orbits such as ES-L1, GEO, HEO, and SSO have the highest success rates, nearing 100%. VLEO and LEO also demonstrate high success rates, though slightly lower. In contrast, launches to ISS, GTO, and SO orbits have lower success rates, with GTO being the least successful. This indicates that while certain orbits benefit from more refined processes and frequent launches, more complex orbits present greater challenges, leading to lower success rates.



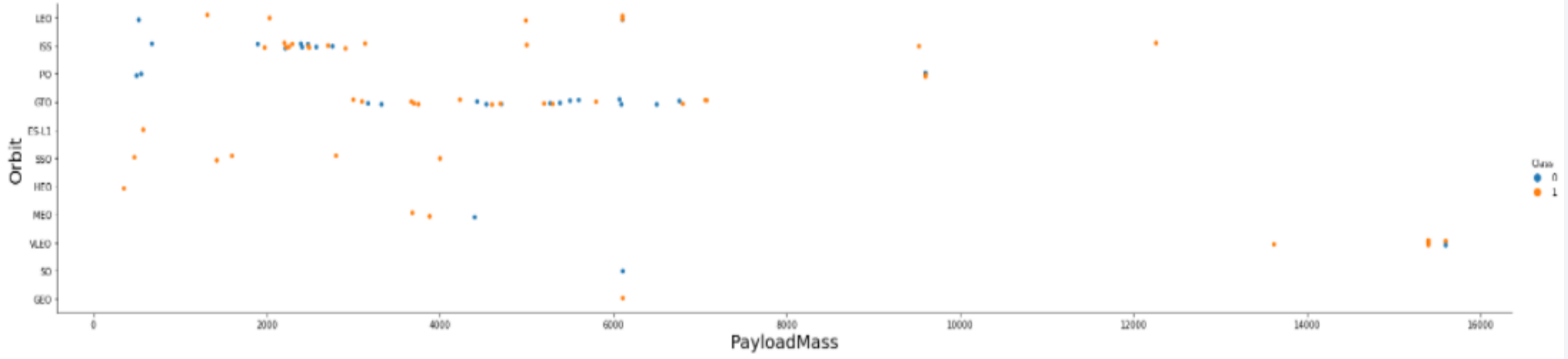
Flight Number vs. Orbit Type

- The scatter plot indicates that orbits such as LEO, ISS, and SSO have higher launch frequencies with increasing success rates over time. While GTO shows mixed results due to its complexity, recent flights demonstrate improved success. Orbits like ES-L1 and GEO, despite fewer launches, maintain high success rates. Overall, the trend suggests that higher flight experience leads to better success rates across various orbit types.



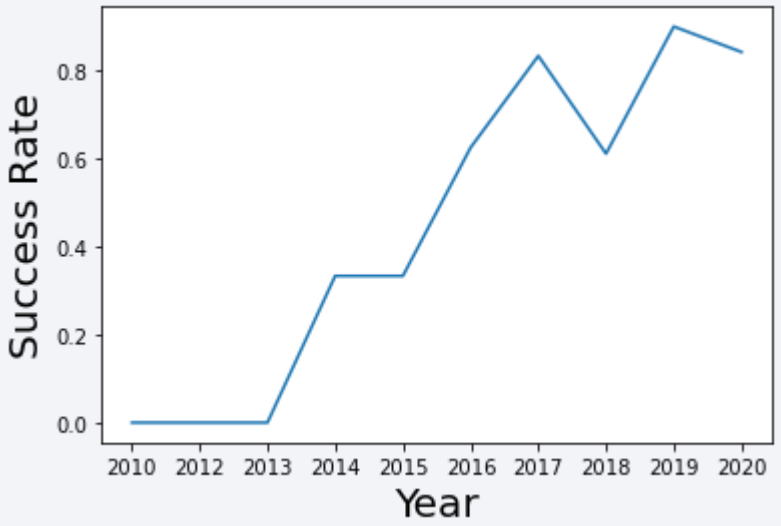
Payload vs. Orbit Type

- The scatter plot shows that lower payload masses (below 6,000 kg) are more frequently launched across various orbits, with both successful and unsuccessful landings. Orbits like LEO, ISS, and PO have a higher success rate even with moderate payloads, while GTO shows mixed outcomes due to its complexity. Although fewer launches carry heavier payloads, successful landings are still achieved, indicating SpaceX's growing capability to handle larger payloads reliably across different orbits.



Launch Success Yearly Trend

- The line plot shows a clear upward trend in SpaceX's success rate over the years. From near-zero success between 2010 and 2013, the rate steadily increased from 2014 onward, with significant growth between 2015 and 2017. Despite a slight dip in 2018, the success rate peaked above 85% in 2019, reflecting continuous improvements and increased reliability in SpaceX’s launch operations over time.



All Launch Site Names

- This query retrieves all unique launch site names from the SPACEXTBL table. The result would provide a list of all distinct launch sites where SpaceX Falcon 9 rockets were launched, ensuring no duplicates in the output.

```
%%sql
SELECT DISTINCT LAUNCH_SITE
FROM SPACEXTBL;
```

```
 * ibm_db_sa://xcg80731:***@ba99a9e6-d59e-4883-8fc0-d6a8c9f7a08f.c1ogj3sd0tgtu0lqde00.databa
Done.
```

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

Launch Site Names Begin with 'CCA'

- This query filters the LAUNCH_SITE column in the SPACEXTBL table for entries starting with CCA and returns the first 5 records. This helps in quickly identifying launch sites located at Cape Canaveral Air Force Station (CCAFS), which is commonly abbreviated as CCA.

Task 2

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

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

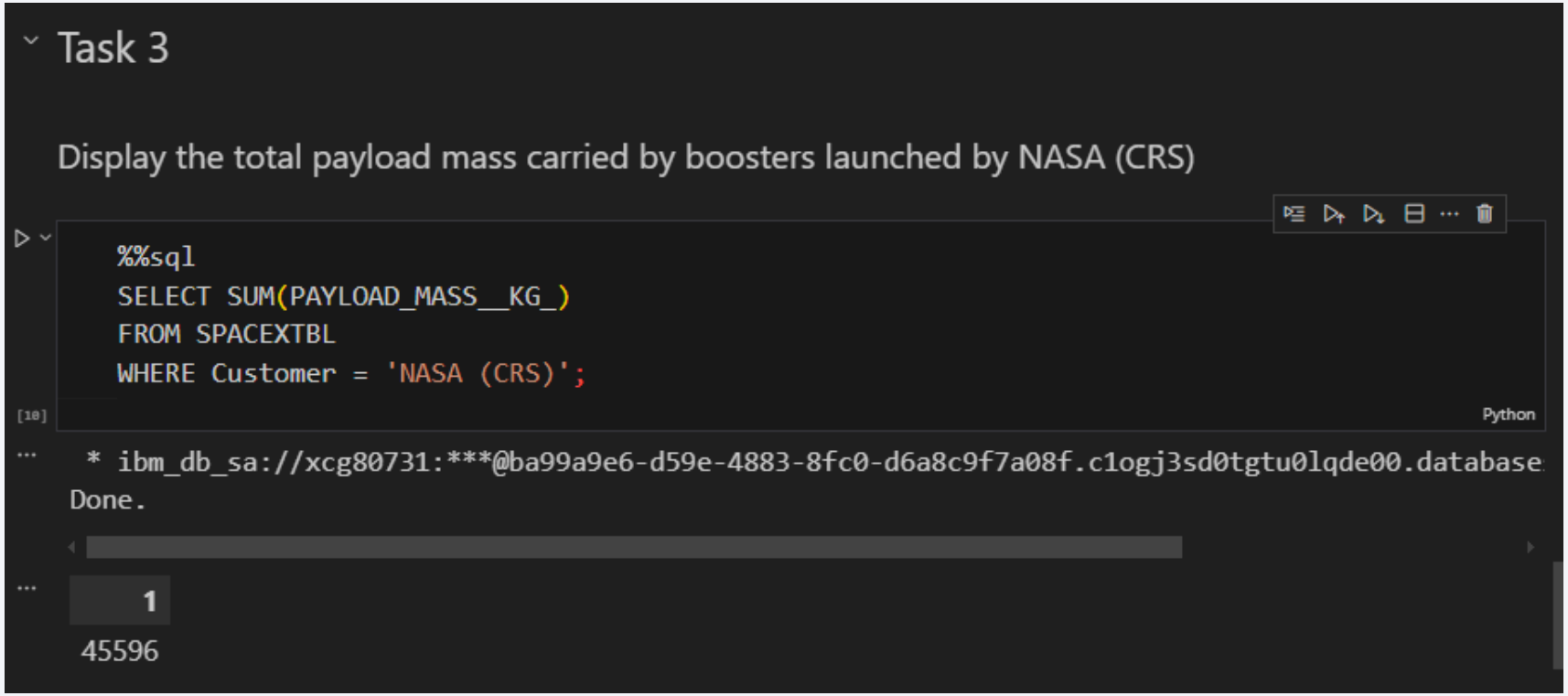
Python

```
 * ibm_db_sa://xcg80731:***@ba99a9e6-d59e-4883-8fc0-d6a8c9f7a08f.c1ogj3sd0tgtu0lqde00.database
Done.
```

launch_site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

Total Payload Mass

- This query sums up the PAYLOAD_MASS__KG_ column for all records where the Customer is 'NASA (CRS)', providing the total payload mass carried by SpaceX boosters for NASA missions.



Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1
- Present your query result with a short explanation here

First Successful Ground Landing Date

- This query retrieves the earliest (MIN(DATE)) successful landing outcome on a ground pad from the SPACEXTBL table, giving the date of the first successful ground landing.

Successful Drone Ship Landing with Payload between 4000 and 6000

- This query filters the SPACEXTBL table for booster versions with successful landings on a drone ship and payload masses greater than 4000 kg but less than 6000 kg, providing the names of boosters meeting these criteria.

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

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

[33] Python

```
 * ibm_db_sa://xcg80731:***@ba99a9e6-d59e-4883-8fc0-d6a8c9f7a08f.c1ogj3sd0tgtu0lqde00.database
Done.
```

booster_version
F9 FT B1021.1
F9 FT B1023.1
F9 FT B1029.2
F9 FT B1038.1
F9 B4 B1042.1
F9 B4 B1045.1
F9 B5 B1046.1

Total Number of Successful and Failure Mission Outcomes

- These queries count the total number of records in the SPACEXTBL table where the MISSION_OUTCOME is either Success or Failure, providing the total counts of successful and failed missions.

Task 7

List the total number of successful and failure mission outcomes

```
%%sql
SELECT MISSION_OUTCOME, COUNT(MISSION_OUTCOME) AS TOTAL_NUMBER
FROM SPACEXTBL
GROUP BY MISSION_OUTCOME;
```

[44] Python

```
 * ibm_db_sa://xcg80731:***@ba99a9e6-d59e-4883-8fc0-d6a8c9f7a08f.c1ogj3sd0tgtu0lqde00.database
Done.
```

mission_outcome	total_number
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

Boosters Carried Maximum Payload

- This query retrieves the unique booster versions from the SPACEXTBL table where the PAYLOAD_MASS__KG_ matches the highest payload mass recorded, providing the names of boosters that carried the maximum payload.

Task 8

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

```
%%sql
SELECT DISTINCT BOOSTER_VERSION
FROM SPACEXTBL
WHERE PAYLOAD_MASS__KG_ = (
    SELECT MAX(PAYLOAD_MASS__KG_)
    FROM SPACEXTBL);
```

Python

```
 * ibm_db_sa://xcg80731:***@ba99a9e6-d59e-4883-8fc0-d6a8c9f7a08f.c1ogj3sd0tgtu0lqde00.database
Done.
```

booster_version
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

2015 Launch Records

- This query retrieves the LANDING__OUTCOME, BOOSTER_VERSION, and LAUNCH_SITE columns from the SPACEXTBL table, filtering for records where the landing outcome was a failure on a drone ship and the launch date was in the year 2015.

Task 9

List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

```
%%sql
SELECT LANDING__OUTCOME, BOOSTER_VERSION, LAUNCH_SITE
FROM SPACEXTBL
WHERE Landing__Outcome = 'Failure (drone ship)'
    AND YEAR(DATE) = 2015;
```

Python

```
 * ibm_db_sa://xcg80731:***@ba99a9e6-d59e-4883-8fc0-d6a8c9f7a08f.c1ogj3sd0tgtu0lqde00.database
Done.
```

landing__outcome	booster_version	launch_site
Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

- This query counts the number of each LANDING__OUTCOME (such as Failure (drone ship) or Success (ground pad)) from the SPACEXTBL table within the specified date range and ranks them in descending order of frequency.

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

```
%%sql
SELECT LANDING__OUTCOME, COUNT(LANDING__OUTCOME) AS TOTAL_NUMBER
FROM SPACEXTBL
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY LANDING__OUTCOME
ORDER BY TOTAL_NUMBER DESC
```

Python

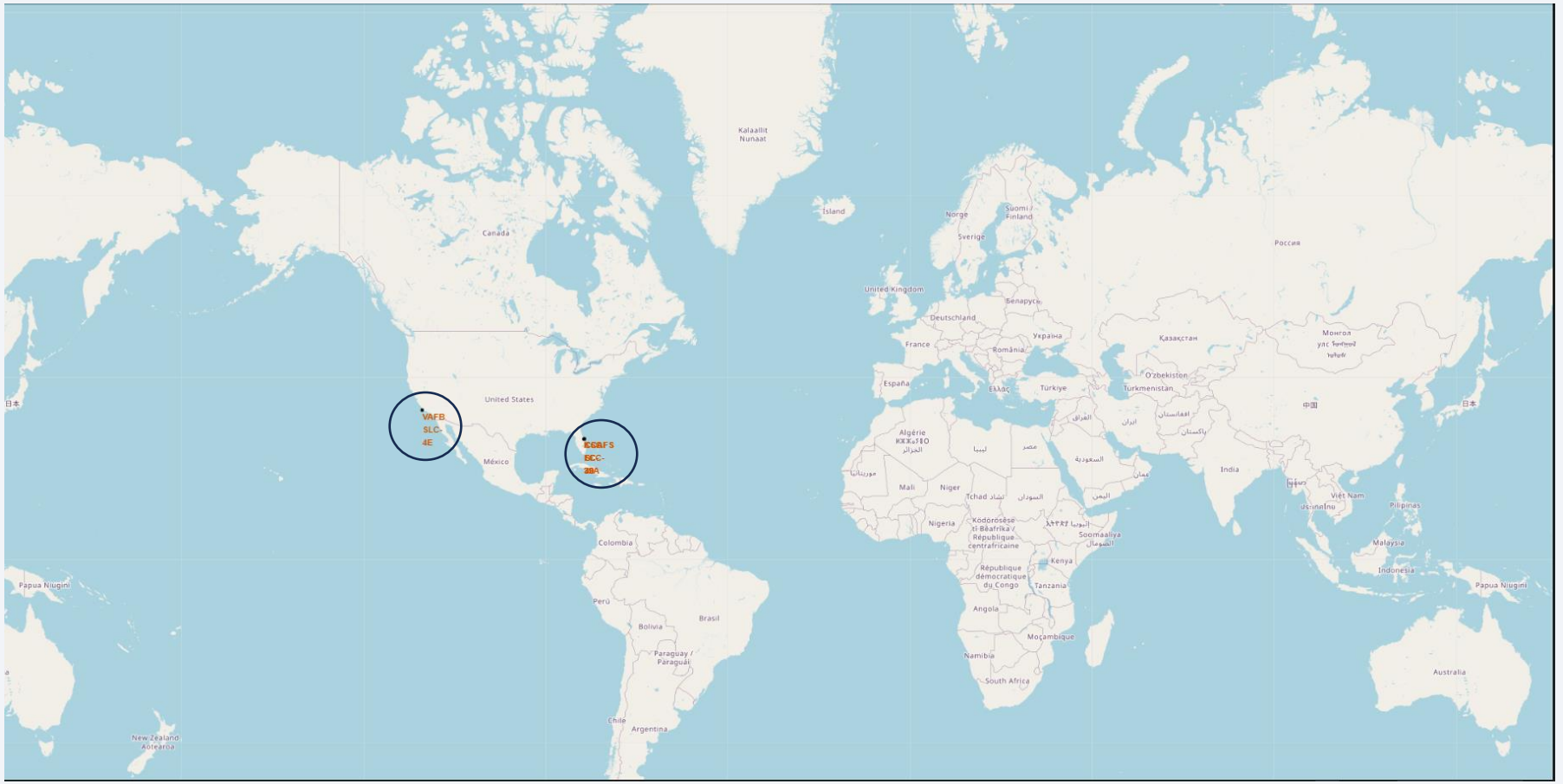
```
 * ibm_db_sa://xcg80731:***@ba99a9e6-d59e-4883-8fc0-d6a8c9f7a08f.c1ogj3sd0tgtu0lqde00.database
Done.
```

landing__outcome	total_number
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

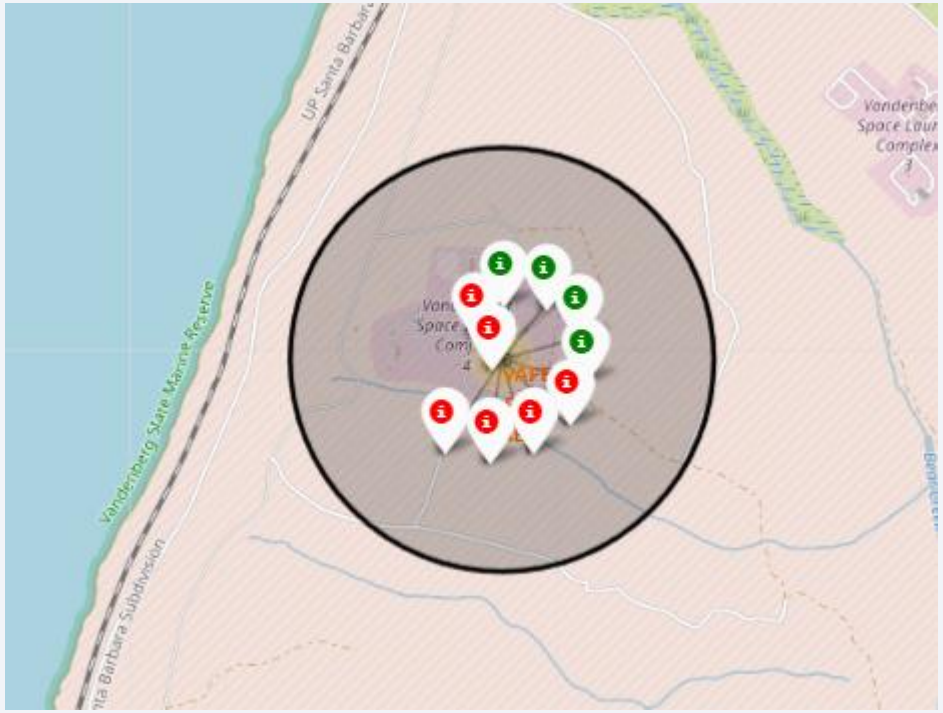
Section 3

Launch Sites Proximities Analysis

All launch sites' location markers on a global map



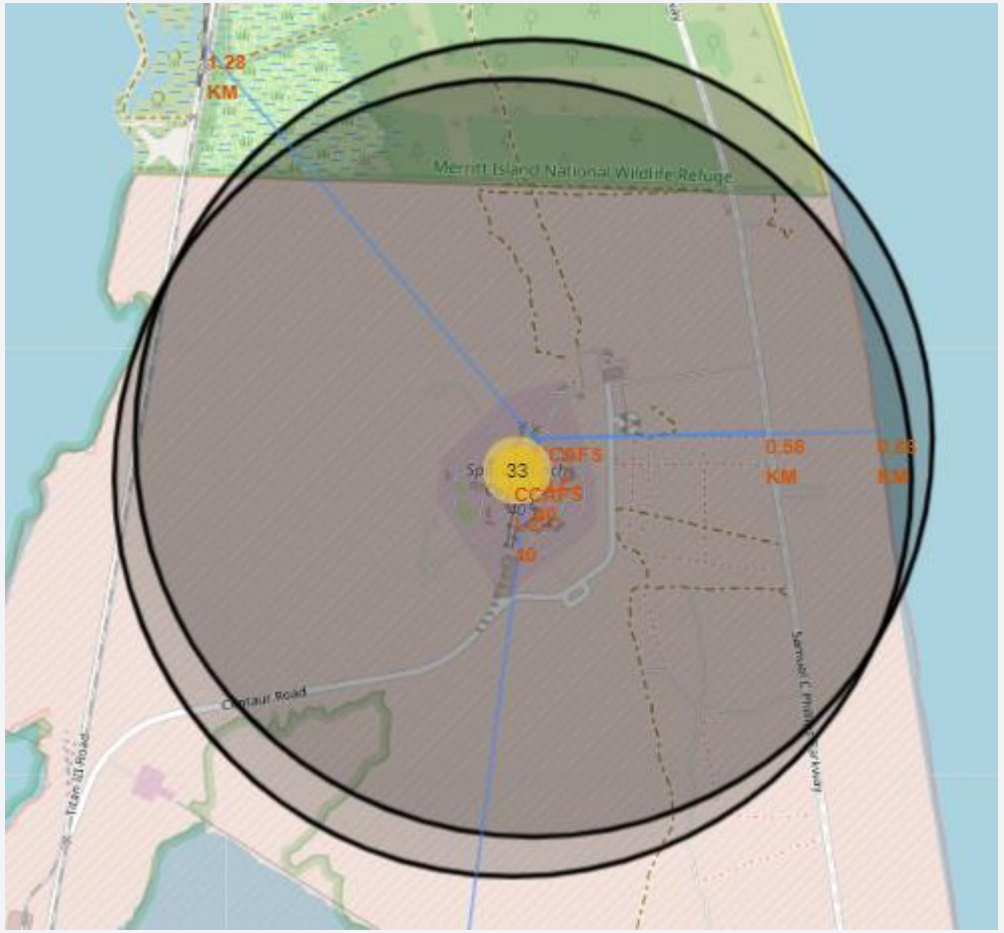
Color-labeled launch outcomes on the map



California

Florida

Closeness to different structures



- Are launch sites in close proximity to railways? Yes
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

Closeness to different structures

```
# Create a marker with distance to a closest city, railway, highway relative to CCAFS S
# Draw a line between the marker to the launch site
closest_highway = 28.56335, -80.57085
closest_railroad = 28.57206, -80.58525
closest_city = 28.10473, -80.64531
```

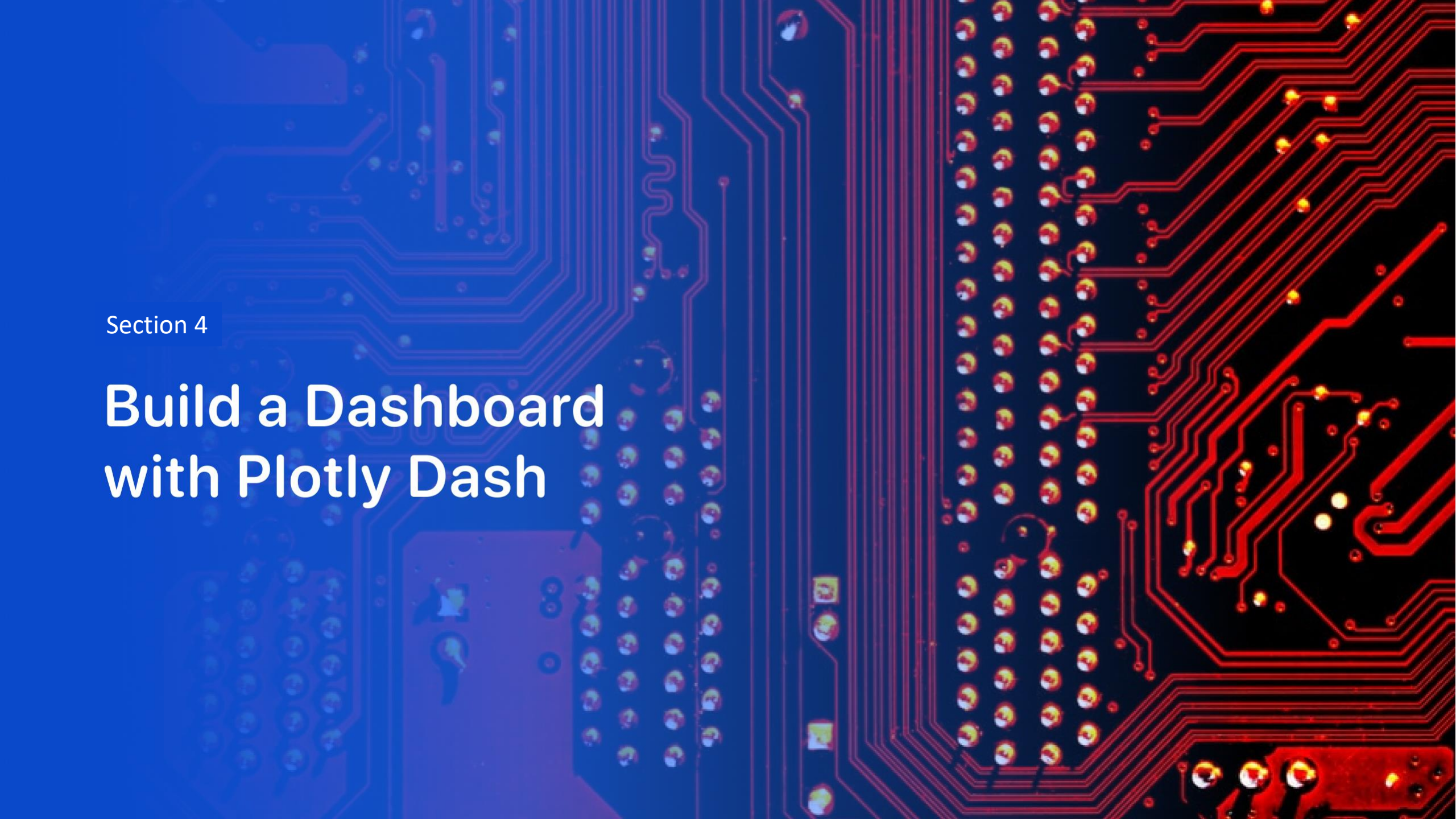
Python

```
distance_highway = calculate_distance(launch_site_lat, launch_site_lon, closest_highway
print('distance_highway =',distance_highway, ' km')
distance_railroad = calculate_distance(launch_site_lat, launch_site_lon, closest_railro
print('distance_railroad =',distance_railroad, ' km')
distance_city = calculate_distance(launch_site_lat, launch_site_lon, closest_city[0], c
print('distance_city =',distance_city, ' km')
```

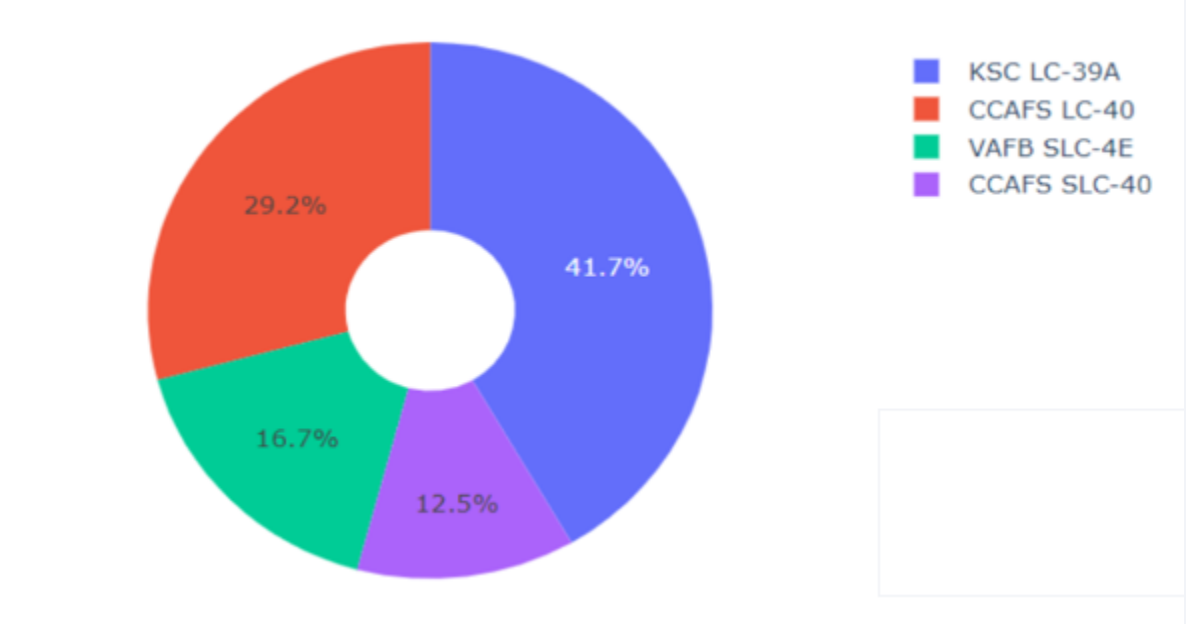
Python

```
distance_highway = 0.5834695366934144  km
distance_railroad = 1.2845344718142522  km
distance_city = 51.43416999517233  km
```

- Launch sites are strategically located near the equator to reduce fuel consumption using Earth's rotation, close to coastlines for safer launch trajectories over the ocean, near highways and railways for efficient transportation of personnel and heavy cargo, and away from densely populated cities to minimize risk to large populations.

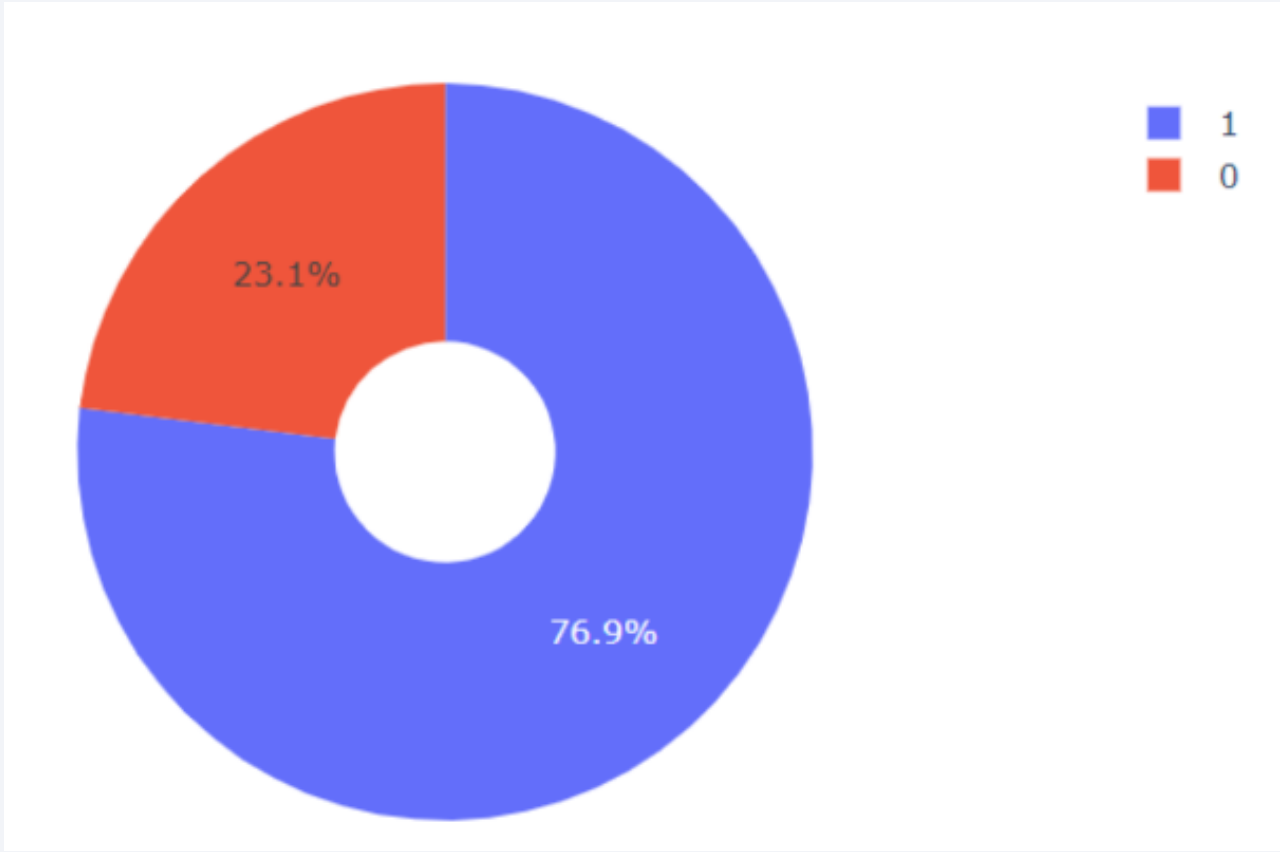

Section 4
Build a Dashboard with Plotly Dash

Piechart - launch success count for all sites



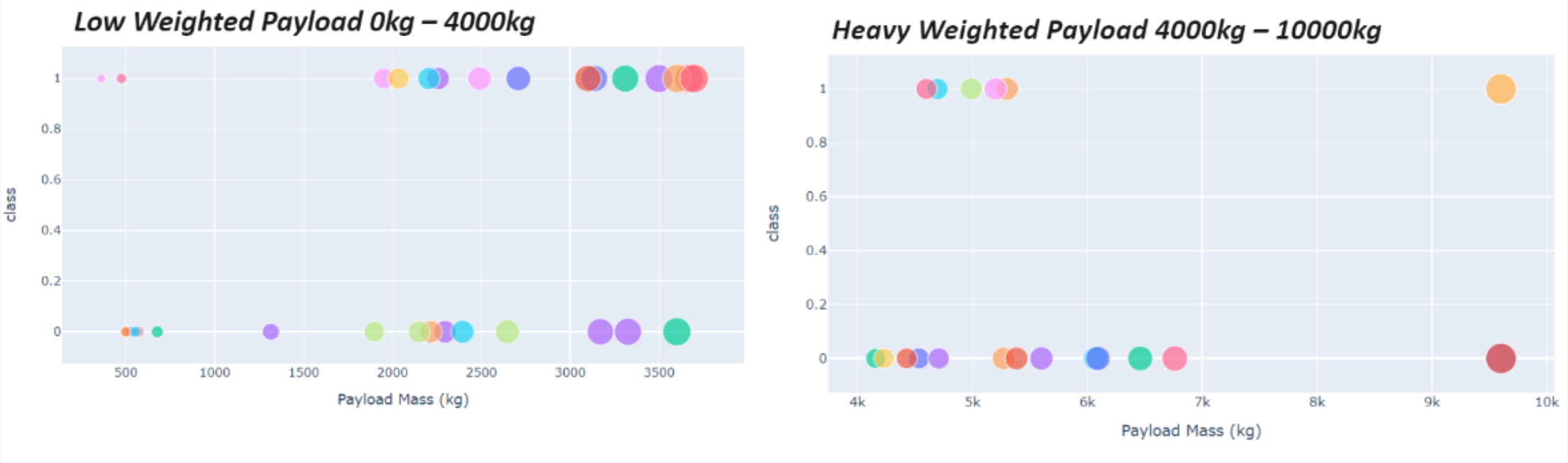
- The most successful launch corresponds to KSC LC-40

Piechart for the launch site with highest launch success ratio



- The launch KSC LC-391 had 76.9% of success.

Payload vs. Launch Outcome scatter plot for all sites

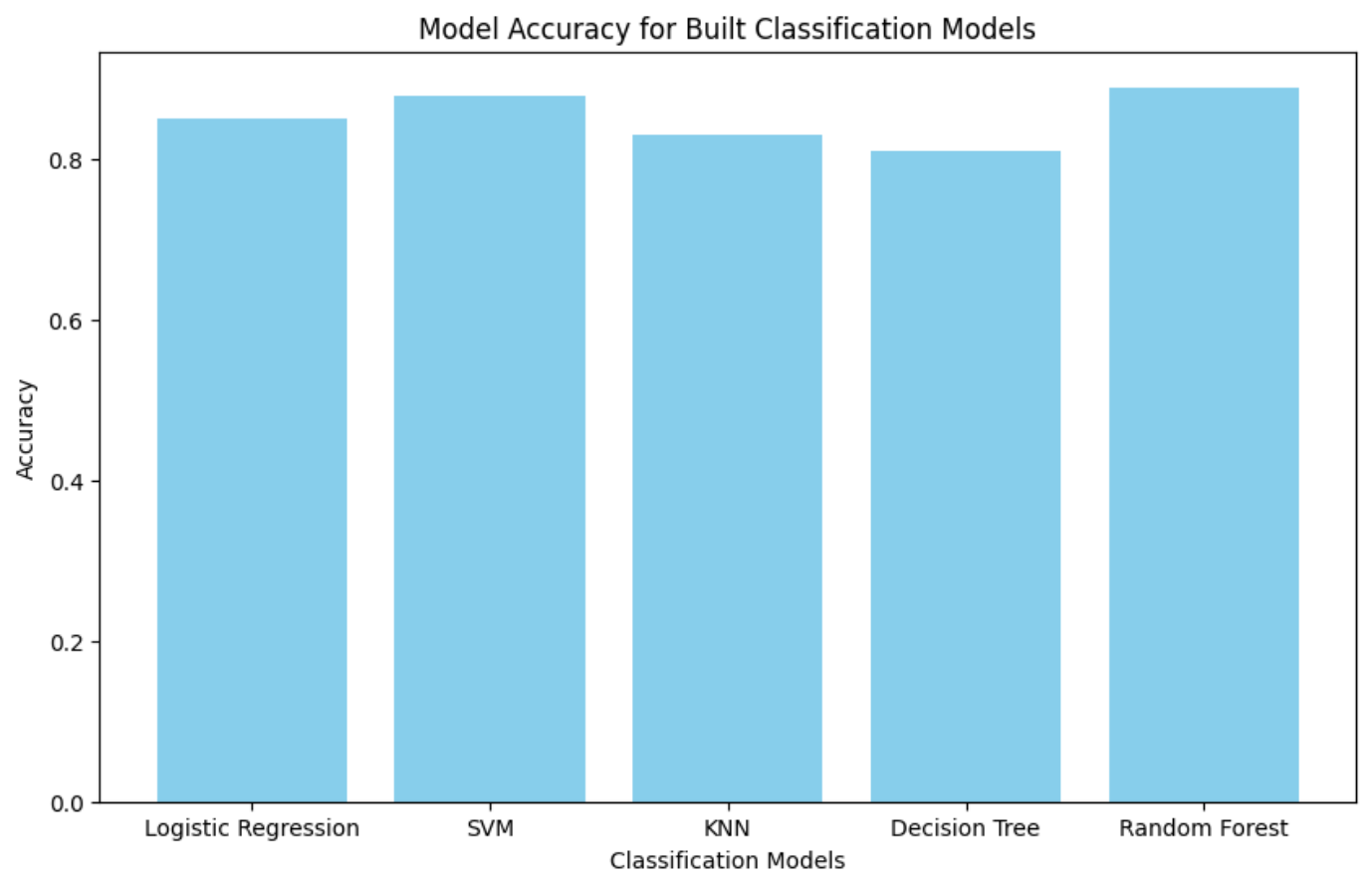


Low weighted success rate is higher than heavy weighted payloads.

Section 5

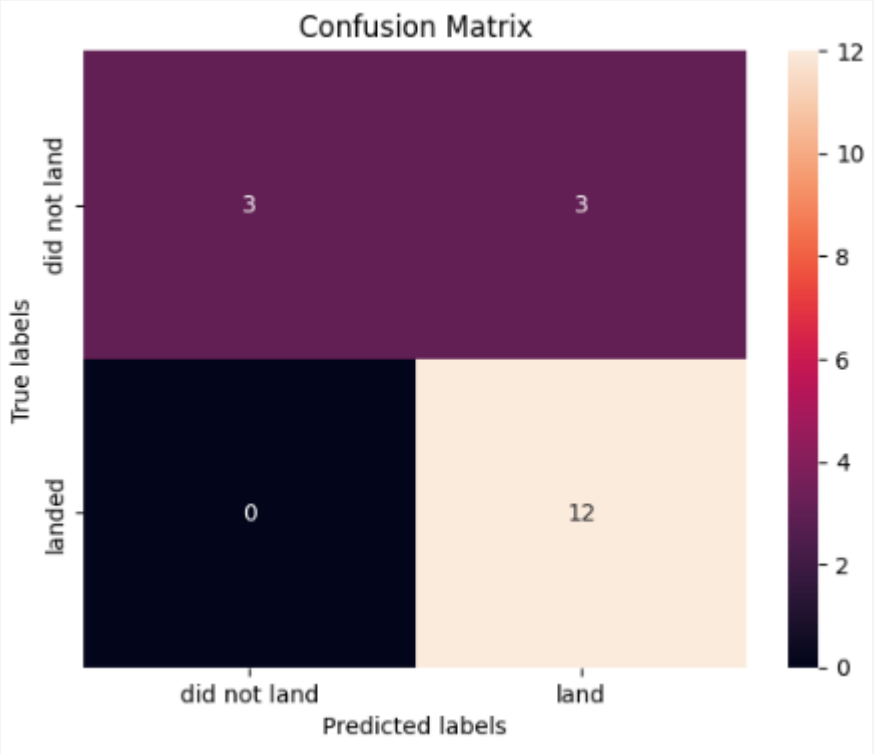
Predictive Analysis (Classification)

Classification Accuracy



Confusion Matrix

- The best performing model is the model with the highest classification accuracy is Random Forest with an accuracy of 0.89.



Conclusions

- 1. The machine learning models developed in the report achieved an accuracy of 83.3% in predicting whether the SpaceX Falcon 9 first stage booster would successfully land. This level of accuracy enables SpaceY to estimate when SpaceX will reuse a booster, thus impacting the cost of their launches.
- 2. SpaceX's public statements indicate that the cost of building a first stage booster is upwards of $15 million. When this cost is factored into SpaceX's standard launch price of $62 million, any mission requiring the sacrifice of the booster could drive the bid to $77 million or more, giving SpaceY a competitive edge when bidding.
- 3. The report recommends that SpaceY could further improve its predictive capabilities by retraining the best-performing model with the entire dataset, which could enhance performance but would sacrifice the ability to measure accuracy using a test set. Additionally, incorporating new launch data as it becomes available would ensure more precise predictions over time.
- 4. Future opportunities include building separate models to predict (1) whether SpaceX will attempt to land a booster and (2) whether the attempt will succeed. Another potential model could predict if SpaceX will reuse a previously flown booster, further refining SpaceY's bidding strategy.

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project

Thank you!