IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

- Presented by:
Mourad El Azhari

- Date: 23 November 2024

Outline

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

Executive Summary

- This capstone project focuses on SpaceX's Falcon 9 rocket launches, aiming to predict the likelihood of a successful first-stage landing.
- This information is vital for determining cost-effectiveness in space missions.
-
- **Key methodologies:**
- Data collection using APIs and web scraping.
- Data wrangling and exploratory data analysis (EDA).
- Machine learning prediction models for landing success.
- Interactive visual analytics using Folium and Plotly Dash.
- The project delivers actionable insights into the factors affecting launch success, providing a foundation for further exploration.

Introduction

- SpaceX offers Falcon 9 rocket launches at a reduced cost due to the reusability of its

- first-stage boosters. The objective is to predict whether the first stage will land
- successfully based on various features.

Key questions addressed:

- - What factors determine landing success?
- - What operating conditions optimize success rates?

Section 1

Methodology

Methodology

- Executive Summary
- Data collection methodology:
  - Describe how data was collected
- Perform data wrangling
  - Describe how data was processed
- 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 Methodology

- To gather the necessary data for this project, I utilized a combination of API calls and web scraping techniques to ensure comprehensive coverage of SpaceX's Falcon 9 launch records.

Data Retrieval via SpaceX API:

- I initiated the data collection by making GET requests to the SpaceX API. The responses, formatted in JSON, were processed and normalized using the .json() function and pandas.json_normalize() to convert the structured data into a pandas DataFrame. This format facilitated easier manipulation and analysis of the data.

Data Cleaning and Preprocessing:

- After loading the data, I performed a thorough cleaning process. Missing values were identified and appropriately handled to maintain data integrity. This step ensured that all analyses and models derived from the dataset were accurate and reliable.

Web Scraping for Additional Records:

- To supplement the API data, I employed web scraping techniques using the BeautifulSoup library. Falcon 9 launch records were extracted from Wikipedia, where I targeted HTML tables containing the necessary information. These tables were parsed, cleaned, and transformed into pandas DataFrames for future integration with the API data.

Data Collection – SpaceX API

- To collect the data, I utilized the SpaceX API by making GET requests to retrieve detailed launch records. The response, provided in JSON format, was parsed and transformed into a pandas DataFrame using the json_normalize() function. This allowed for efficient data manipulation and analysis.
- Once the data was imported, I performed essential cleaning steps to address inconsistencies and missing values. This included identifying gaps in the dataset, filling in missing entries where applicable, and ensuring proper formatting. Basic data wrangling techniques were also applied to structure the data effectively for further analysis and visualization.
- Through this methodical process, I prepared the dataset to be both accurate and ready for subsequent steps in the project.
- **The GitHub URL is:**
- https://github.com/Azaricode/IBM_data_science_capstone/blob/main/Data_Collection_API.ipynb

1. Get request for rocket launch data using API

In [6]:
```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

In [7]:
```
response = requests.get(spacex_url)
```

2. Use json_normalize method to convert json result to dataframe

In [12]:
```
# Use json_normalize method to convert the json result into a dataframe

# decode response content as json
static_json_df = res.json()
```

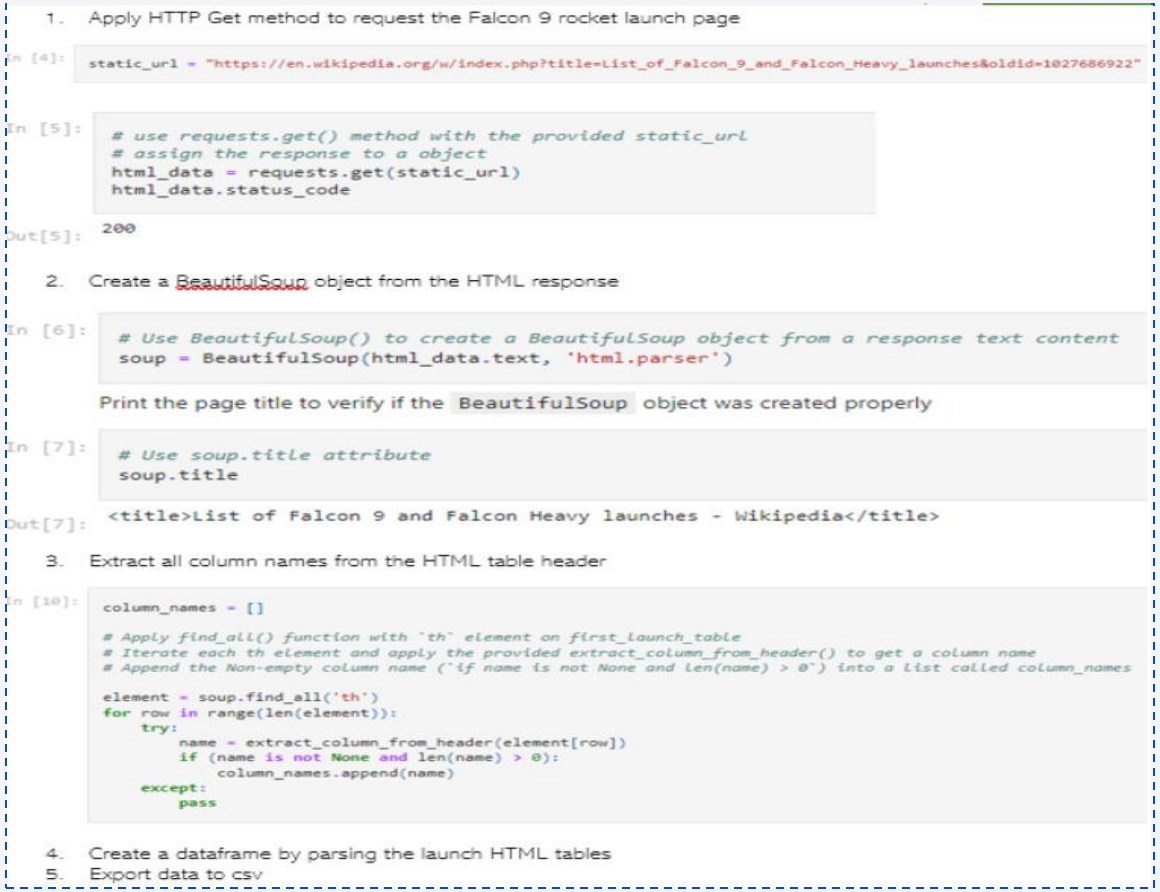
In [13]:
```
# apply json_normalize
data = pd.json_normalize(static_json_df)
```

3. We then performed data cleaning and filling in the missing values

In [30]:
```
rows = data_falcon9['PayloadMass'].values.tolist()[0]

df_rows = pd.DataFrame(rows)
df_rows = df_rows.replace(np.nan, PayloadMass)

data_falcon9['PayloadMass'][0] = df_rows.values
data_falcon9
```

Data Collection - Scraping

- For this project, I utilized web scraping techniques to extract Falcon 9 launch records from publicly available online resources. Using the BeautifulSoup library, I navigated the HTML structure of the target webpage and located the specific table containing the launch records.
- The data from the table was carefully parsed and transformed into a pandas DataFrame for easier manipulation and analysis. This process ensured that the extracted information was well-organized and ready for integration with other datasets, enabling a more comprehensive analysis of the Falcon 9 launches.
- The GitHub URL is :
- https://github.com/Azaricode/IBM_data_science_capstone/blob/main/Data_Collection_with_Web_Scraping.ipynb



1. Apply HTTP Get method to request the Falcon 9 rocket launch page

```
In [4]: static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

```
In [5]: # use requests.get() method with the provided static_url
        # assign the response to a object
        html_data = requests.get(static_url)
        html_data.status_code
Out[5]: 200
```

2. Create a BeautifulSoup object from the HTML response

```
In [6]: # Use BeautifulSoup() to create a BeautifulSoup object from a response text content
        soup = BeautifulSoup(html_data.text, 'html.parser')
```

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

```
In [7]: # Use soup.title attribute
        soup.title
Out[7]: <title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

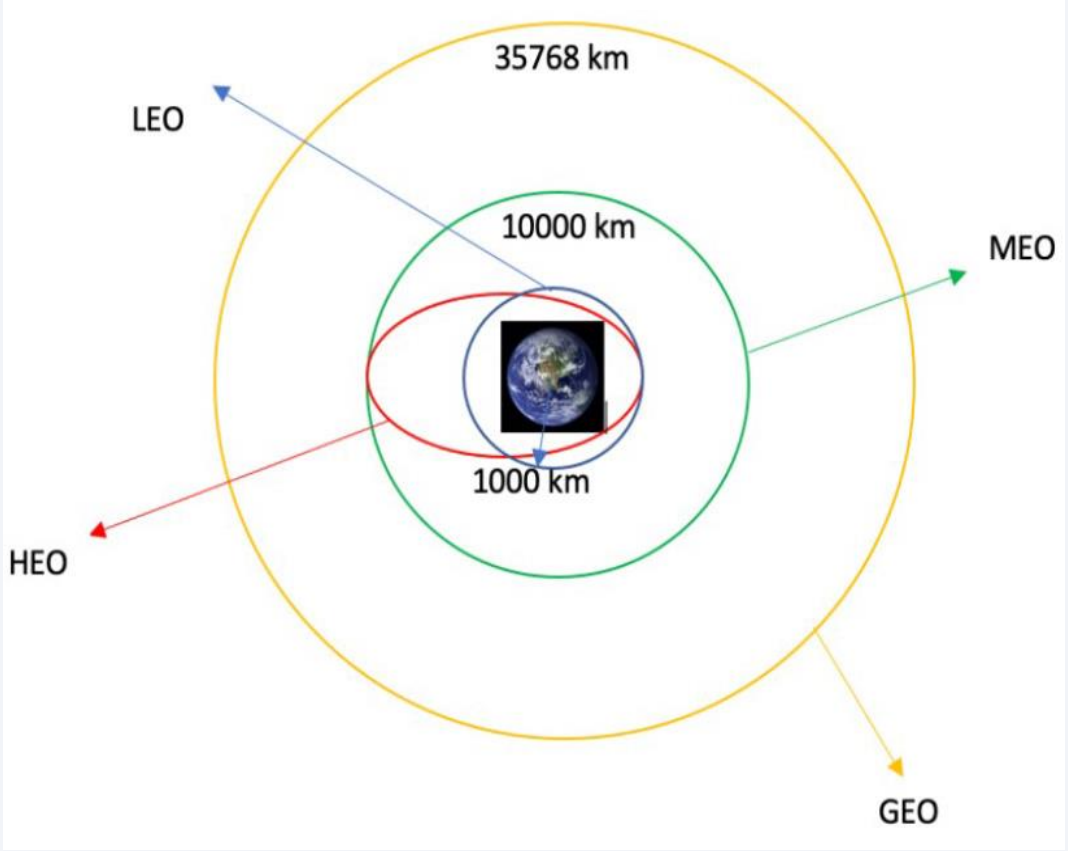
3. Extract all column names from the HTML table header

```
In [10]: column_names = []

# Apply find_all() function with `th` element on first_launch_table
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names

element = soup.find_all('th')
for row in range(len(element)):
    try:
        name = extract_column_from_header(element[row])
        if (name is not None and len(name) > 0):
            column_names.append(name)
    except:
        pass
```

4. Create a dataframe by parsing the launch HTML tables
5. Export data to csv

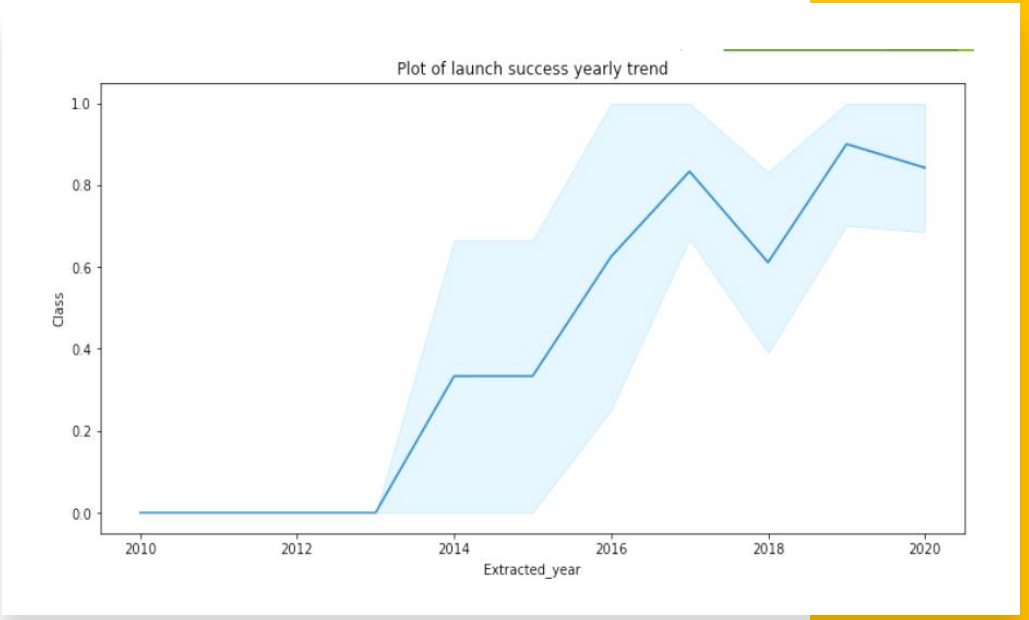
Data Wrangling

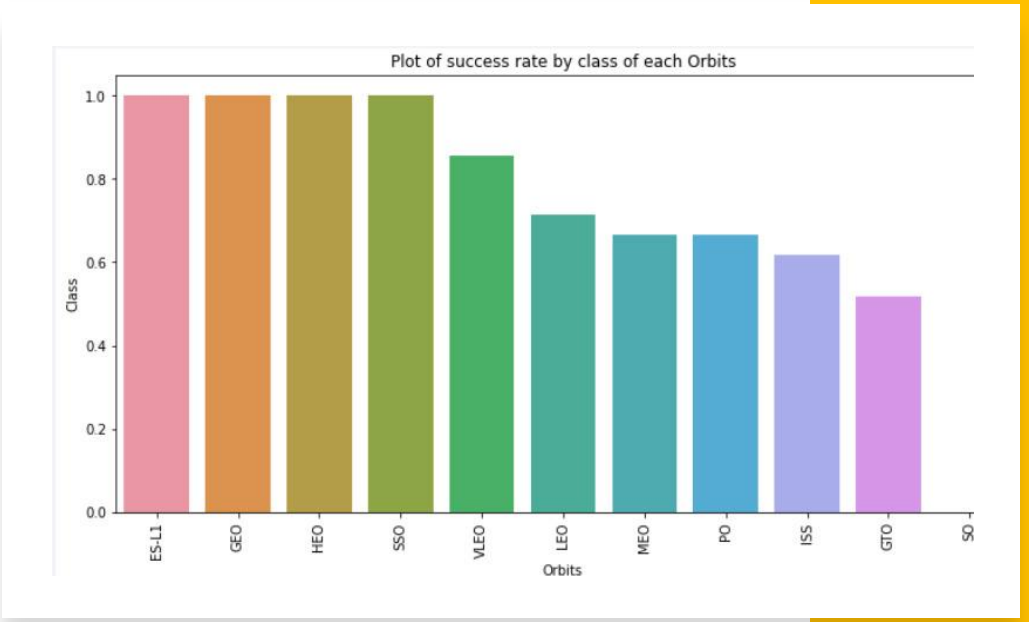
- As part of the project, I conducted exploratory data analysis (EDA) to uncover patterns and insights within the dataset. A critical step involved determining the training labels by analyzing key features of the data.
- First, I calculated the number of launches at each site and analyzed the frequency and distribution of various orbit types. This provided valuable insights into the relationship between launch sites, orbit types, and their respective outcomes.
- Additionally, I created a new label, *Landing Outcome*, derived from the existing outcome column. This label categorized the landing results, which were then exported to a CSV file for further analysis and use in the predictive modeling phase. These steps ensured that the data was well-prepared for building machine learning models.
- The GitHub URL is:
- https://github.com/Azaricode/IBM_data_science_capstone/blob/main/Data_Wrangling.ipynb



EDA with Data Visualization

- I explored the dataset by visualizing key relationships, such as flight number vs. launch site, payload vs. launch site, and success rates for each orbit type. Additionally, I analyzed the yearly trend of launch successes and the interaction between flight numbers and orbit types to identify patterns and insights.
- The GitHub URL is:
- https://github.com/Azaricode/IBM_data_science_capstone/blob/main/EDA_with_Data_Visualization.ipynb





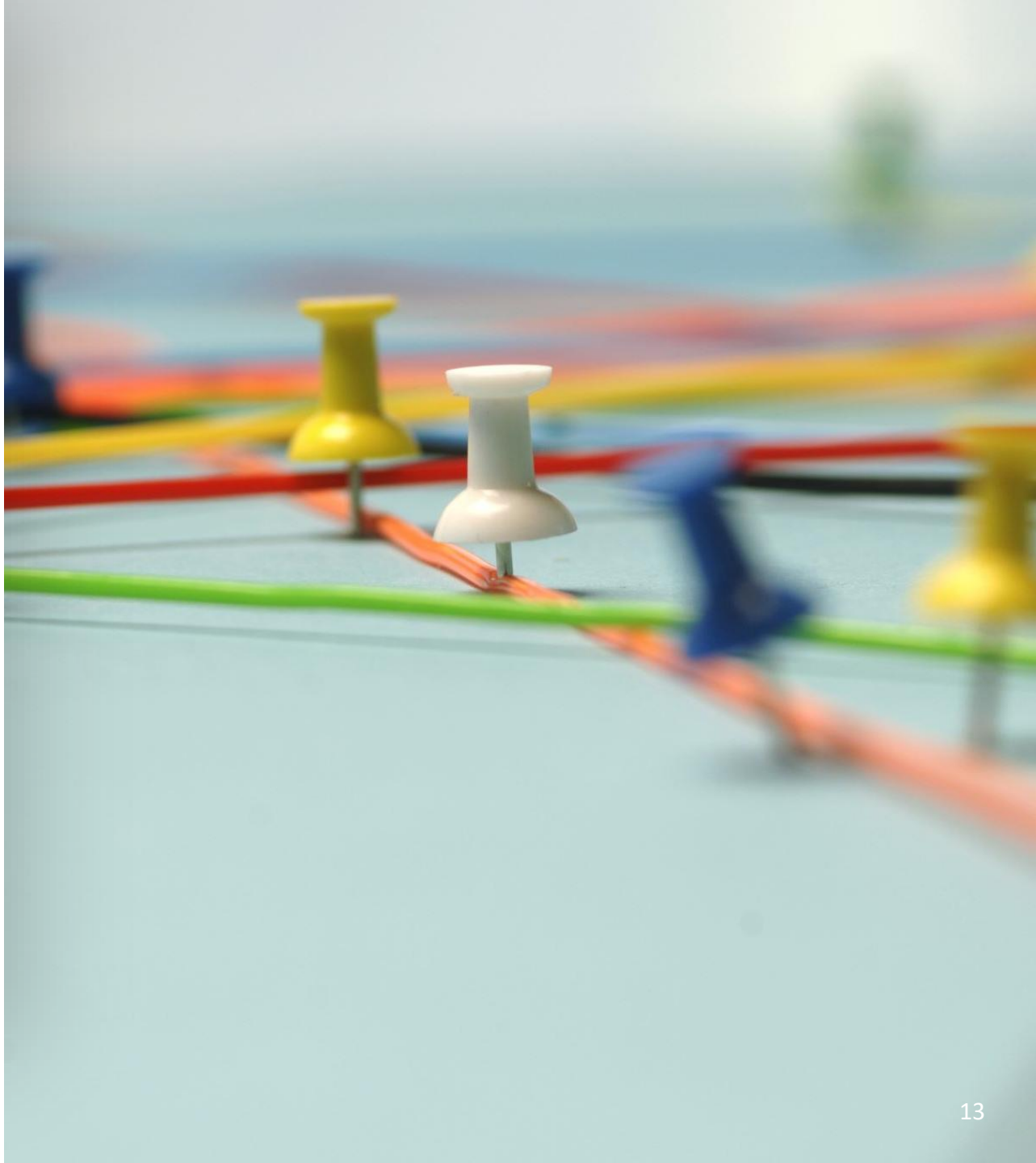
EDA with SQL

- The SpaceX dataset was loaded into a PostgreSQL database directly within the Jupyter Notebook. SQL queries were applied to extract insights, such as:
- Unique launch site names.
- Total payload mass carried by NASA (CRS) boosters.
- Average payload mass for F9 v1.1 boosters.
- Total successful and failed mission outcomes.
- Failed landings on drone ships with corresponding booster versions and launch sites.

- The GitHub URL is:
- tps://github.com/Azaricode/IBM_data_science_capstone/blob/main/EDA_with_SQL.ipynb

Build an Interactive Map with Folium

- Using Folium, I visualized all launch sites by adding map objects like markers, circles, and lines to represent the success or failure of launches. The launch outcomes were classified as 0 (failure) and 1 (success) and displayed with color-coded markers to identify sites with higher success rates.
- Additionally, I calculated distances from launch sites to nearby features such as railways, highways, coastlines, and cities. This analysis answered key questions, including whether launch sites are strategically positioned relative to these proximities.
- The GitHub URL is :
- **https://github.com/Azaricode/IBM_data_science_capstone/blob/main/Interactive_Visual_Analytics_with_Folium.ipynb**



Build a Dashboard with Plotly Dash

- I developed an interactive dashboard using Plotly Dash. The dashboard included pie charts displaying the total launches by site and scatter plots illustrating the relationship between launch outcomes and payload mass (kg) for various booster versions, providing dynamic insights into key factors influencing success rates.
- The GitHub URL is:
- https://github.com/Azaricode/IBM_data_science_capstone/blob/main/Extracting_and_Visualizing_Stock_Data.ipynb

Predictive Analysis (Classification)

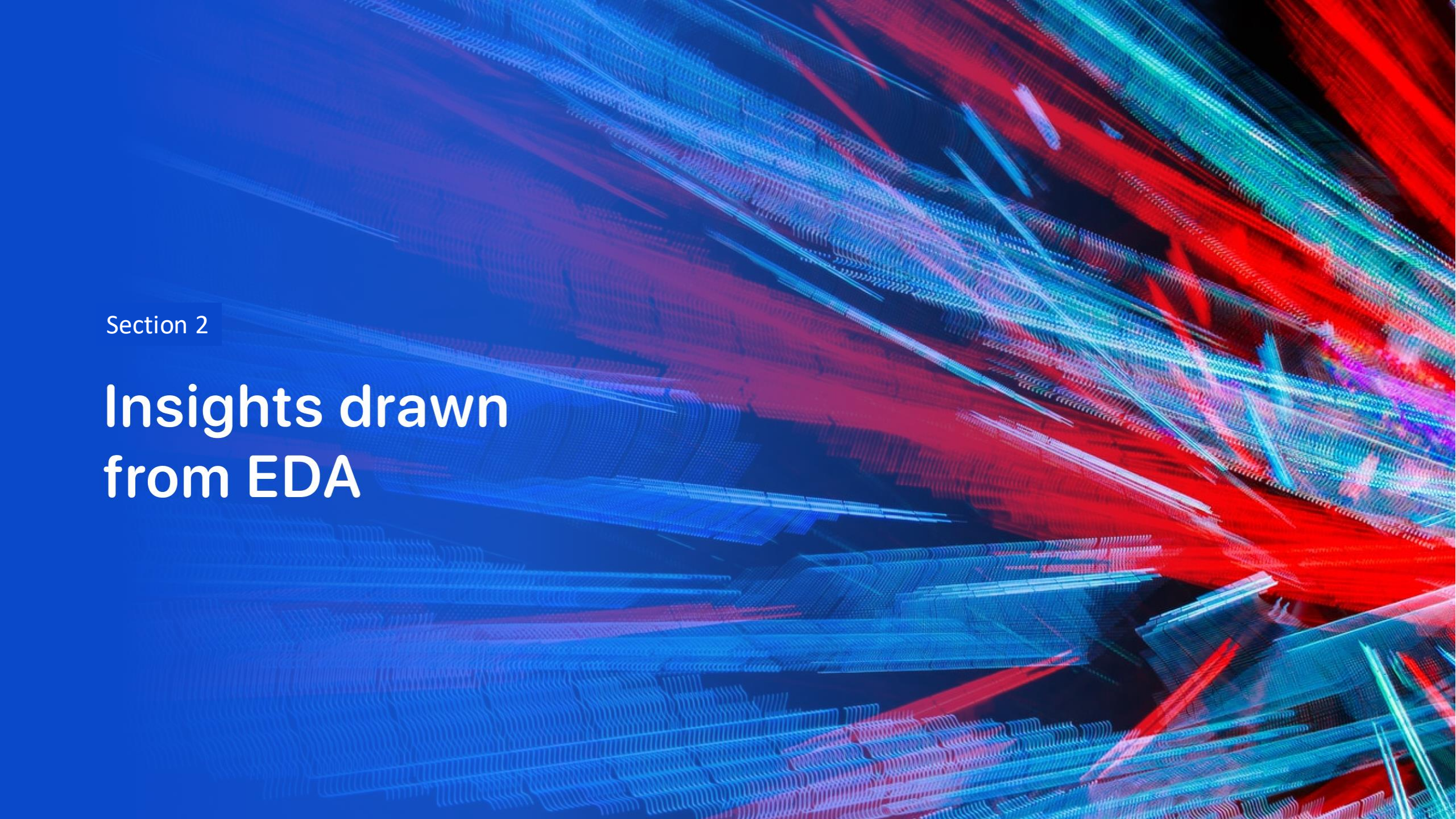
- I used numpy and pandas to load and preprocess the data, transforming it and splitting it into training and testing sets. Various machine learning models were built, and hyperparameters were fine-tuned using GridSearchCV. Accuracy was the primary metric to evaluate performance, which I improved further through feature engineering and algorithm tuning. Ultimately, I identified the best-performing classification model for predicting outcomes.
- The GitHub URL is :
- **https://github.com/Azaricode/IBM_data_science_capstone/blob/main/Machine_Learning_Prediction.ipynb**

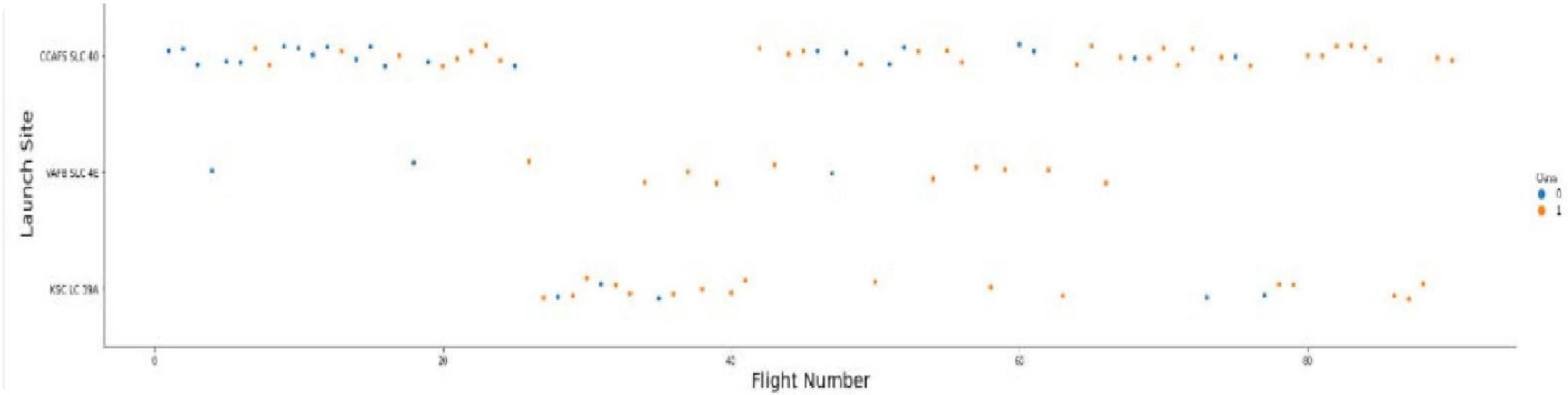
Results

Exploratory Data Analysis (EDA): Highlighted relationships between key variables, including payload, orbit type, and launch success trends.

Interactive Analytics: Demonstrated through visual dashboards and interactive maps, providing insights into launch site performance and proximities.

Predictive Analysis: Successfully identified the most accurate classification model, showcasing its performance in predicting launch outcomes.


Section 2
Insights drawn from EDA

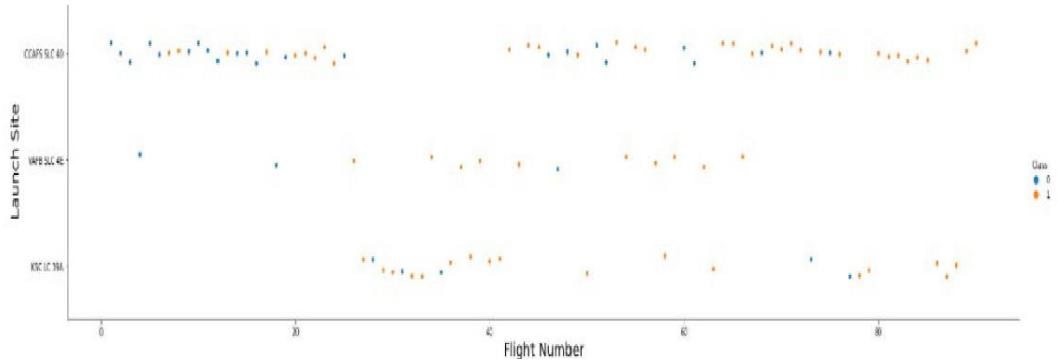


Flight Number vs. Launch Site

- The analysis revealed a positive correlation between the number of flights conducted at a launch site and its success rate, indicating that higher flight frequencies contribute to improved outcomes.

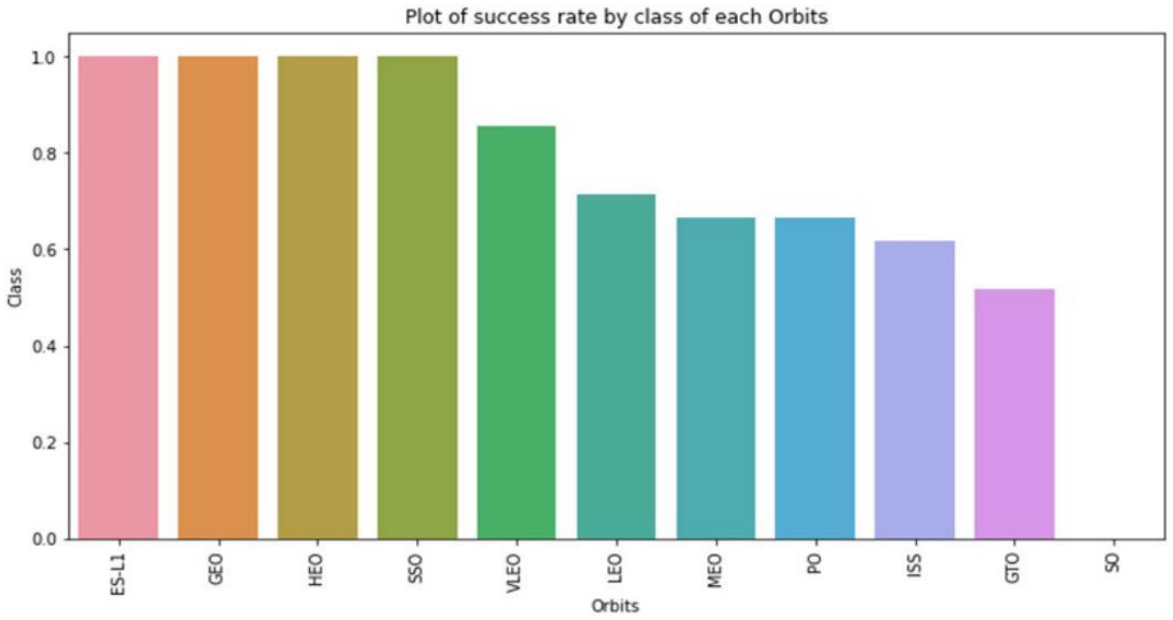
Payload vs. Launch Site

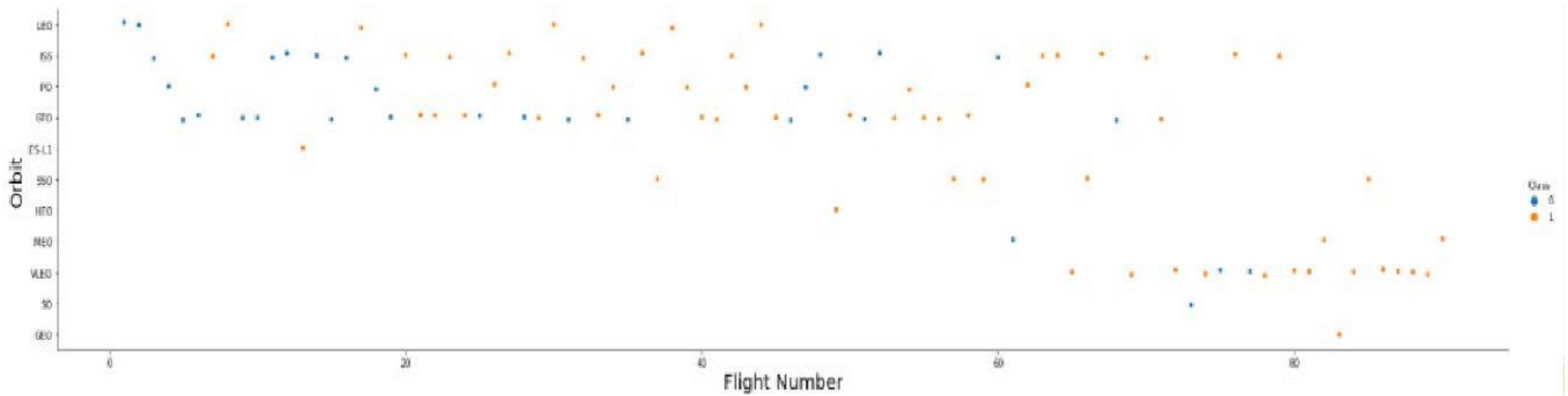
The greater the payload mass for launch site CCAFS SLC 40 the higher the success rate for the rocket.



Success Rate vs. Orbit Type

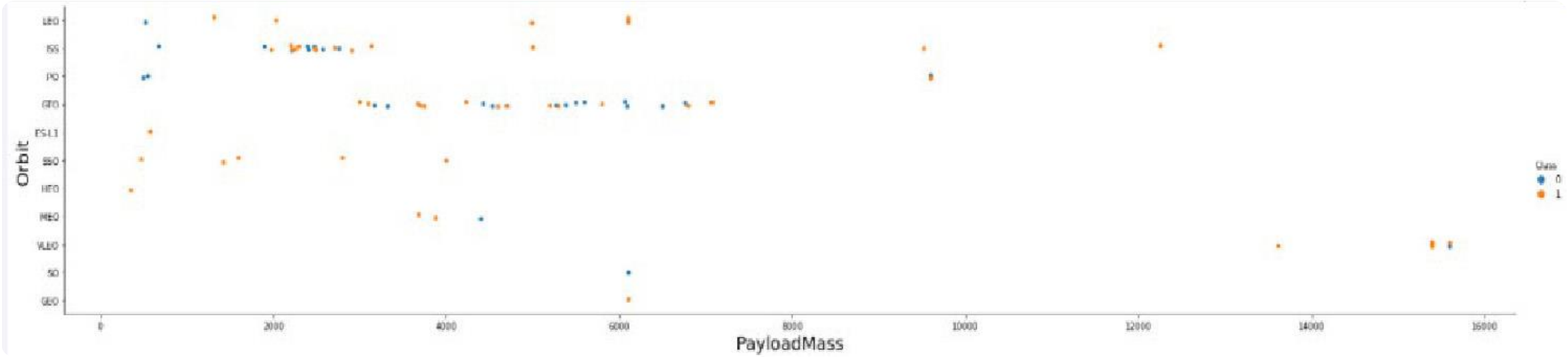
- The visualization indicates that orbits such as ES-L1, GEO, HEO, SSO, and VLEO exhibited the highest success rates, highlighting their reliability for successful launches.





Flight Number vs. Orbit Type

- The plot demonstrates that in the LEO orbit, success rates are positively correlated with the number of flights. In contrast, no such relationship is observed for the GTO orbit, suggesting differing dynamics across orbit types.

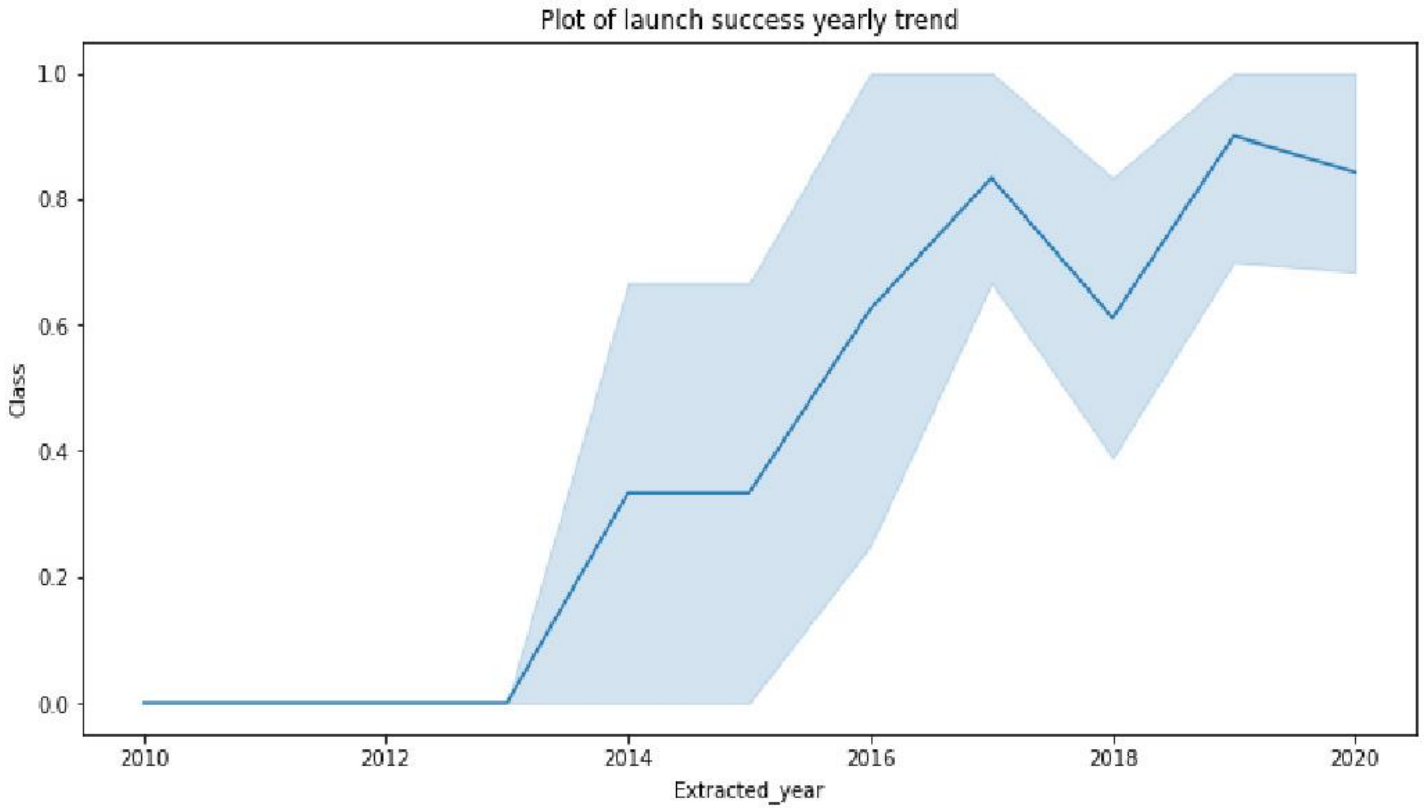


Payload vs. Orbit Type

- The analysis shows that orbits such as PO, LEO, and ISS achieve higher success rates when carrying heavier payloads, indicating their suitability for larger missions.

Launch Success Yearly Trend

- The plot reveals a steady increase in success rates from 2013 to 2020, highlighting significant improvements in operational efficiency over time.



All Launch Site Names

- The keyword DISTINCT was utilized to extract and display only the unique launch site names from the SpaceX dataset, ensuring a concise and accurate representation of the data.

Display the names of the unique launch sites in the space mission

In [10]:
```
task_1 = '''
        SELECT DISTINCT LaunchSite
        FROM SpaceX
'''
create_pandas_df(task_1, database=conn)
```

Out[10]:

	launchsite
0	KSC LC-39A
1	CCAFS LC-40
2	CCAFS SLC-40
3	VAFB SLC-4E

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

- A SQL query was executed to retrieve the first 5 records of launch sites with names starting with CCA, providing targeted insights into these specific locations.

Total Payload Mass

- Using a SQL query, we calculated the total payload carried by NASA boosters to be 45,596, highlighting the significant contribution of these missions.

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

In [12]:
```
task_3 = '''
        SELECT SUM(PayloadMassKG) AS Total_PayloadMass
        FROM SpaceX
        WHERE Customer LIKE 'NASA (CRS)'
        '''
create_pandas_df(task_3, database=conn)
```

Out[12]:

	total_payloadmass
0	45596

Average Payload Mass by F9 v1.1

- The average payload mass carried by the F9 v1.1 booster version was calculated to be 2,928.4 kg, reflecting its payload efficiency.

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

In [13]:
```
task_4 = '''
        SELECT AVG(PayloadMassKG) AS Avg_PayloadMass
        FROM SpaceX
        WHERE BoosterVersion = 'F9 v1.1'
        '''
create_pandas_df(task_4, database=conn)
```

Out[13]:

	avg_payloadmass
0	2928.4

First Successful Ground Landing Date

- The analysis identified that the first successful landing on a ground pad occurred on December 22, 2015, marking a significant milestone in SpaceX's history.

In [14]:

```
task_5 = '''
        SELECT MIN(Date) AS FirstSuccessfull_landing_date
        FROM SpaceX
        WHERE LandingOutcome LIKE 'Success (ground pad)'
        '''
create_pandas_df(task_5, database=conn)
```

Out[14]:

	firstsuccessfull_landing_date
0	2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- Using the WHERE clause, we filtered boosters that successfully landed on a drone ship. Additionally, the AND condition was applied to identify landings with payload masses between 4,000 and 6,000 kg, providing focused insights into these specific missions.

In [15]:
```
task_6 = '''
        SELECT BoosterVersion
        FROM SpaceX
        WHERE LandingOutcome = 'Success (drone ship)'
            AND PayloadMassKG > 4000
            AND PayloadMassKG < 6000
        '''
create_pandas_df(task_6, database=conn)
```

Out[15]:

	boosterversion
0	F9 FT B1022
1	F9 FT B1026
2	F9 FT B1021.2
3	F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

- The wildcard character % was used in a WHERE clause to filter mission outcomes, allowing us to identify records categorized as either successful or failed.

In [16]:

```
task_7a = '''
        SELECT COUNT(MissionOutcome) AS SuccessOutcome
        FROM SpaceX
        WHERE MissionOutcome LIKE 'Success%'
        '''

task_7b = '''
        SELECT COUNT(MissionOutcome) AS FailureOutcome
        FROM SpaceX
        WHERE MissionOutcome LIKE 'Failure%'
        '''
print('The total number of successful mission outcome is:')
display(create_pandas_df(task_7a, database=conn))
print()
print('The total number of failed mission outcome is:')
create_pandas_df(task_7b, database=conn)
```

```
The total number of successful mission outcome is:
```

	successoutcome
0	100

```
The total number of failed mission outcome is:
```

Out[16]:

	failureoutcome
0	1

Boosters Carried Maximum Payload

- To identify the booster carrying the maximum payload, a subquery within the WHERE clause was combined with the MAX() function, effectively isolating the record with the highest payload

In [17]:

```
task_8 = '''
        SELECT BoosterVersion, PayloadMassKG
        FROM SpaceX
        WHERE PayloadMassKG = (
                                SELECT MAX(PayloadMassKG)
                                FROM SpaceX
                                )
        ORDER BY BoosterVersion
        '''
create_pandas_df(task_8, database=conn)
```

Out[17]:

	boosterversion	payloadmasskg
0	F9 B5 B1048.4	15600
1	F9 B5 B1048.5	15600
2	F9 B5 B1049.4	15600
3	F9 B5 B1049.5	15600
4	F9 B5 B1049.7	15600
5	F9 B5 B1051.3	15600
6	F9 B5 B1051.4	15600
7	F9 B5 B1051.6	15600
8	F9 B5 B1056.4	15600
9	F9 B5 B1058.3	15600
10	F9 B5 B1060.2	15600
11	F9 B5 B1060.3	15600

2015 Launch Records

In [18]:
```
task_9 = '''
        SELECT BoosterVersion, LaunchSite, LandingOutcome
        FROM SpaceX
        WHERE LandingOutcome LIKE 'Failure (drone ship)'
            AND Date BETWEEN '2015-01-01' AND '2015-12-31'
        '''
create_pandas_df(task_9, database=conn)
```

Out[18]:

	boosterversion	launchsite	landingoutcome
0	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
1	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

- A combination of WHERE, LIKE, AND, and BETWEEN conditions was used to filter failed drone ship landings in 2015, including the corresponding booster versions and launch site names, providing detailed insights into these specific outcomes.

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

- The landing outcomes and their counts were selected using a SQL query, with the WHERE clause filtering outcomes between 2010-06-04 and 2010-03-20. The GROUP BY clause was applied to categorize outcomes, and the ORDER BY clause sorted them in descending order, providing a clear view of the distribution and frequency of outcomes within this timeframe.

In [19]:
```
task_10 = '''
        SELECT LandingOutcome, COUNT(LandingOutcome)
        FROM SpaceX
        WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
        GROUP BY LandingOutcome
        ORDER BY COUNT(LandingOutcome) DESC
        '''
create_pandas_df(task_10, database=conn)
```

Out[19]:

	landingoutcome	count
0	No attempt	10
1	Success (drone ship)	6
2	Failure (drone ship)	5
3	Success (ground pad)	5
4	Controlled (ocean)	3
5	Uncontrolled (ocean)	2
6	Precluded (drone ship)	1
7	Failure (parachute)	1

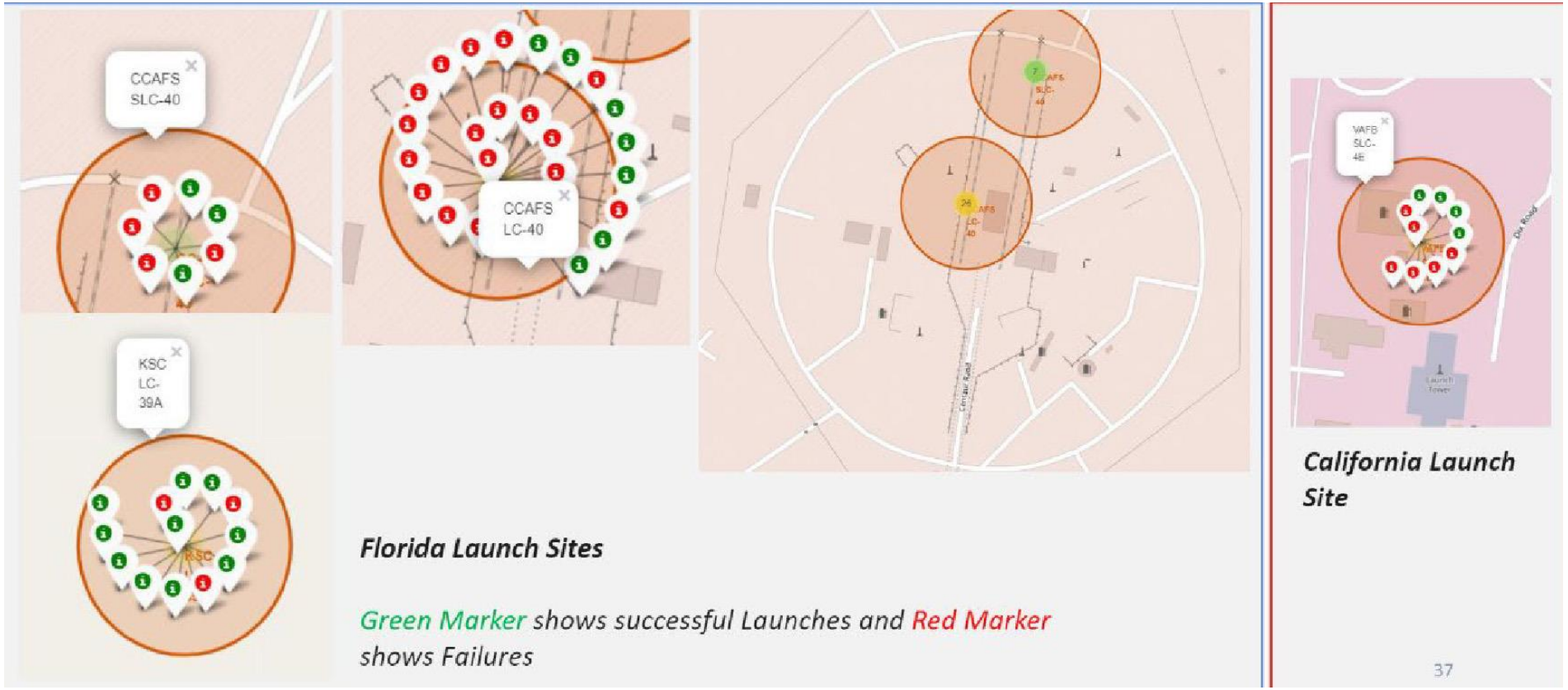
Section 3

Launch Sites Proximities Analysis

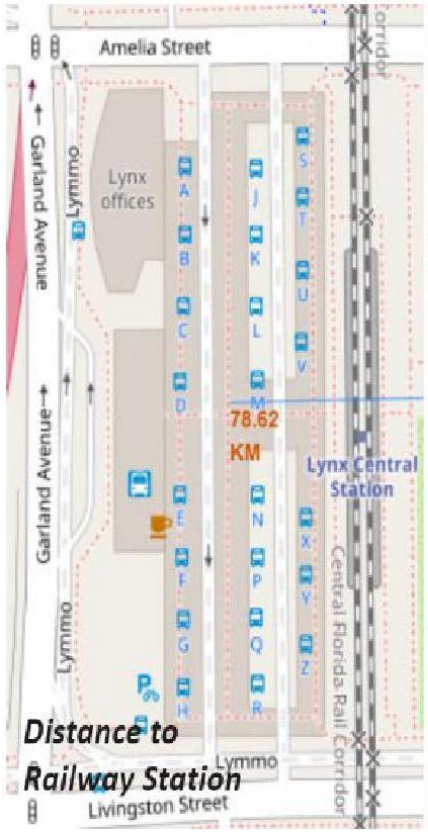
All launch sites global map markers

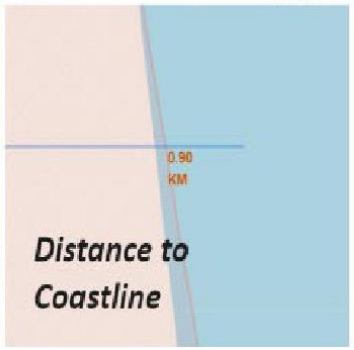
We can see that the SpaceX launch sites are in the United States of America coasts. Florida and California

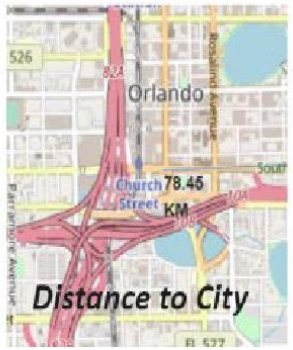
Markers showing launch sites with color labels



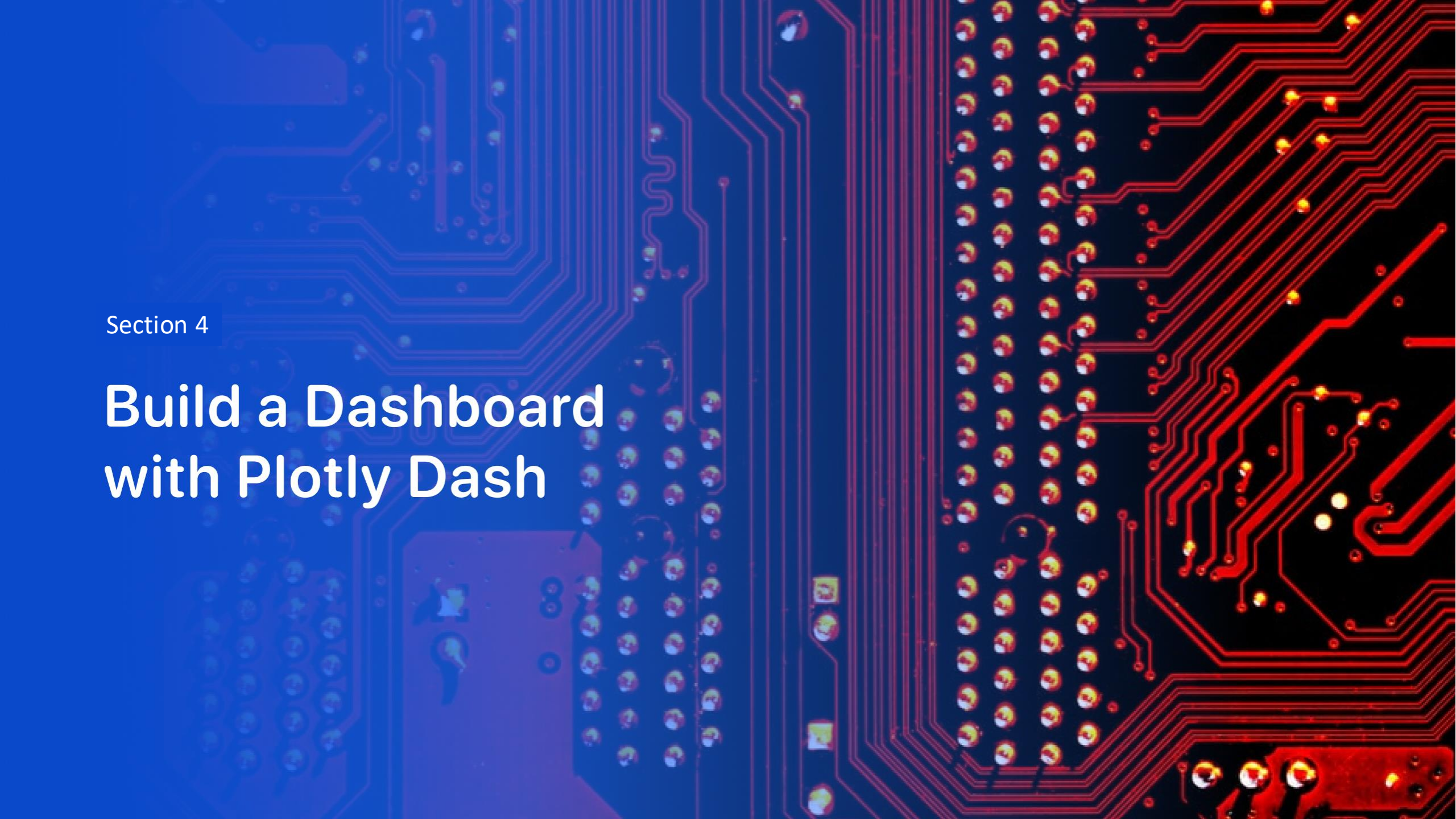
Launch site distance to landmarks



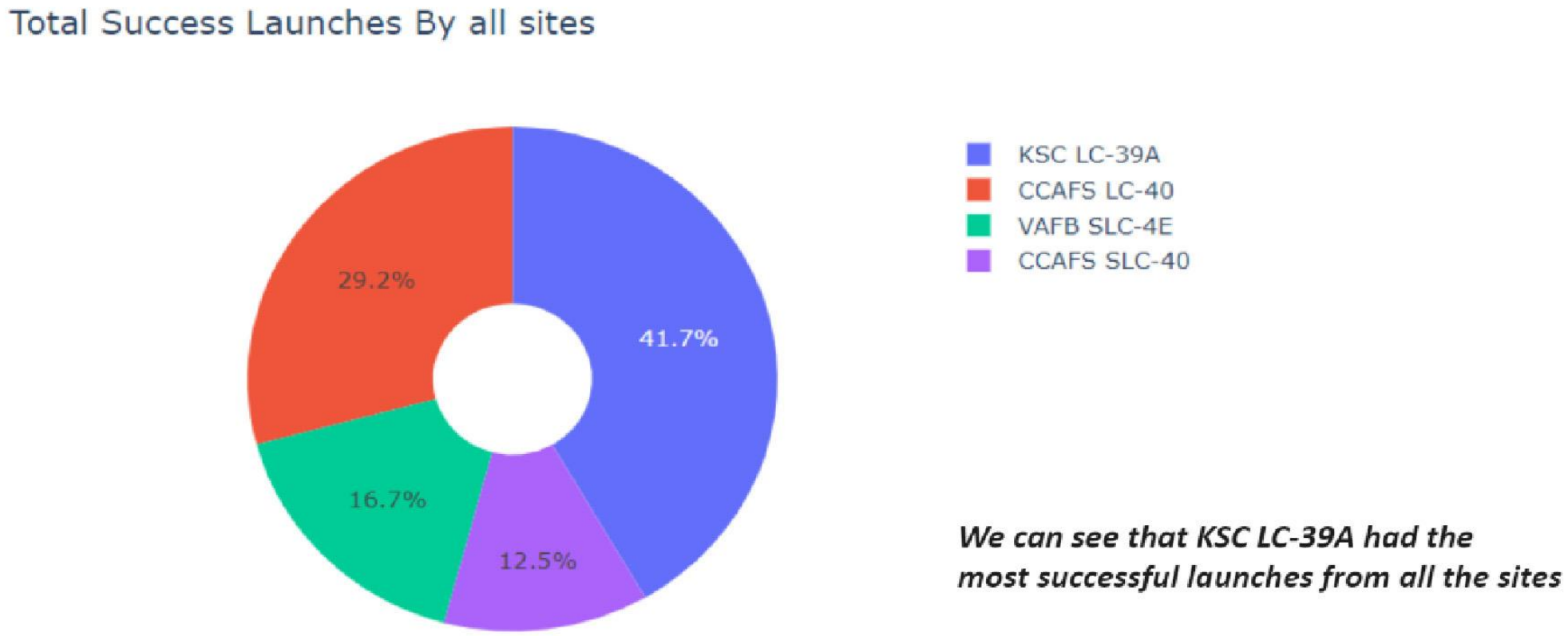




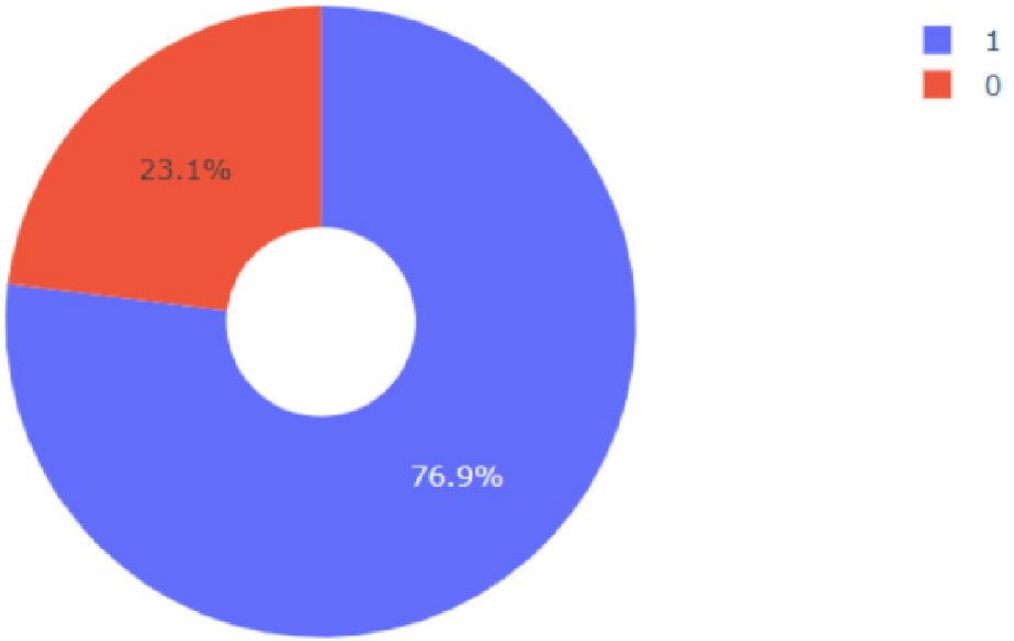
- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes


Section 4
Build a Dashboard with Plotly Dash

Pie Chart Depicting the Success Percentage Achieved by Each Launch Site



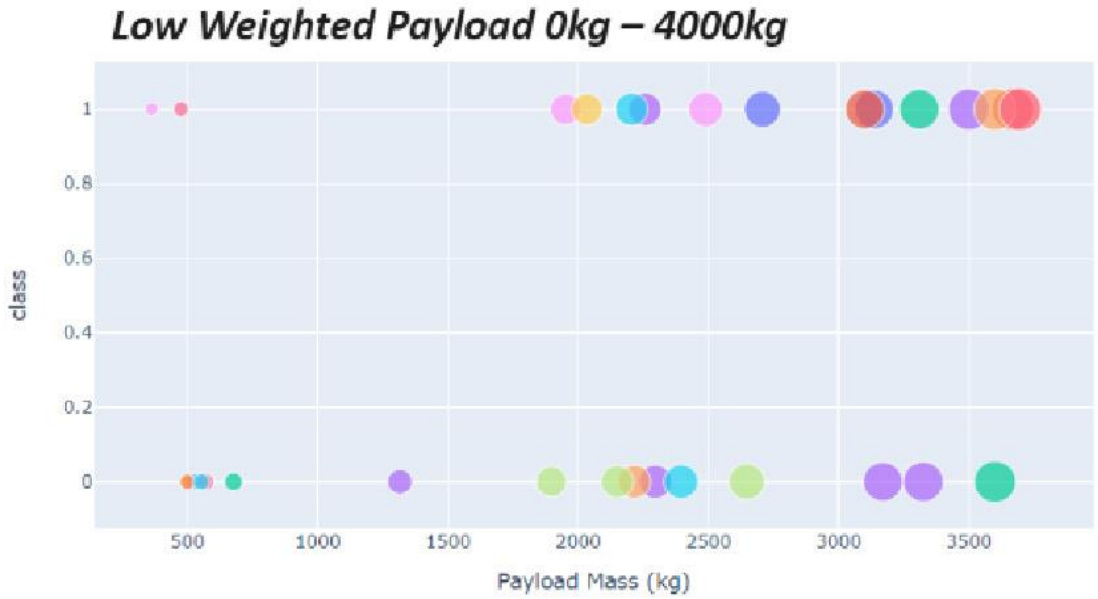
We can see that KSC LC-39A had the most successful launches from all the sites

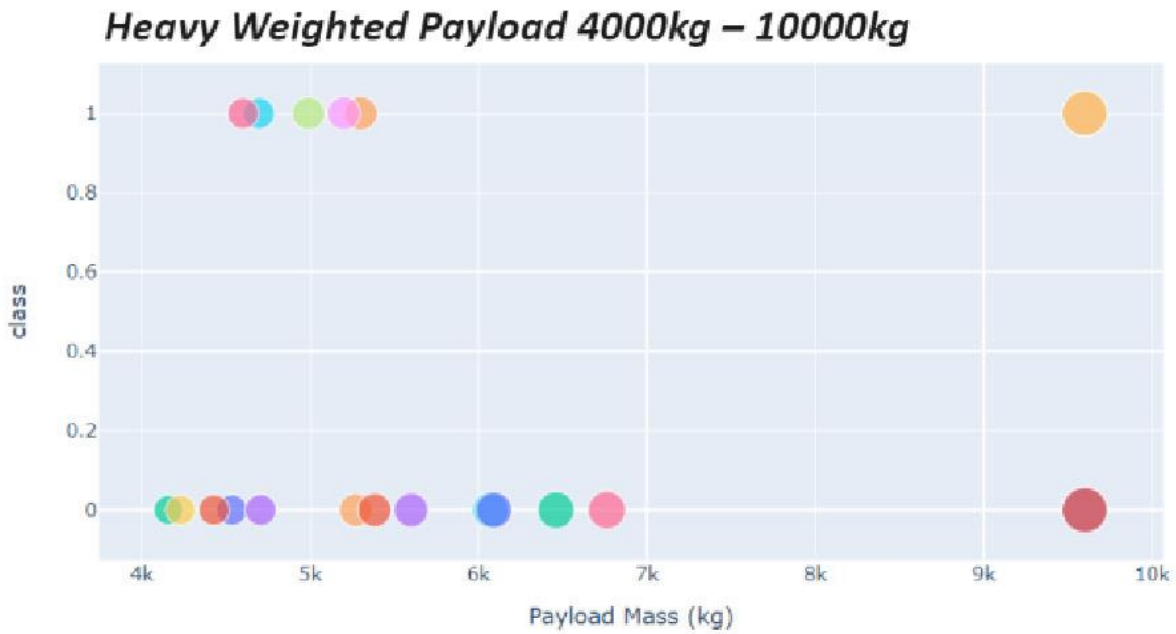


KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

Pie Chart Depicting the Launch Site with the Highest Success Ratio

Scatter Plot of Payload vs. Launch Outcome Across Sites with Range Slider for Payload Selection





We can see the success rates for low weighted payloads is higher than the heavy weighted payloads

Section 5

Predictive Analysis (Classification)

Classification Accuracy

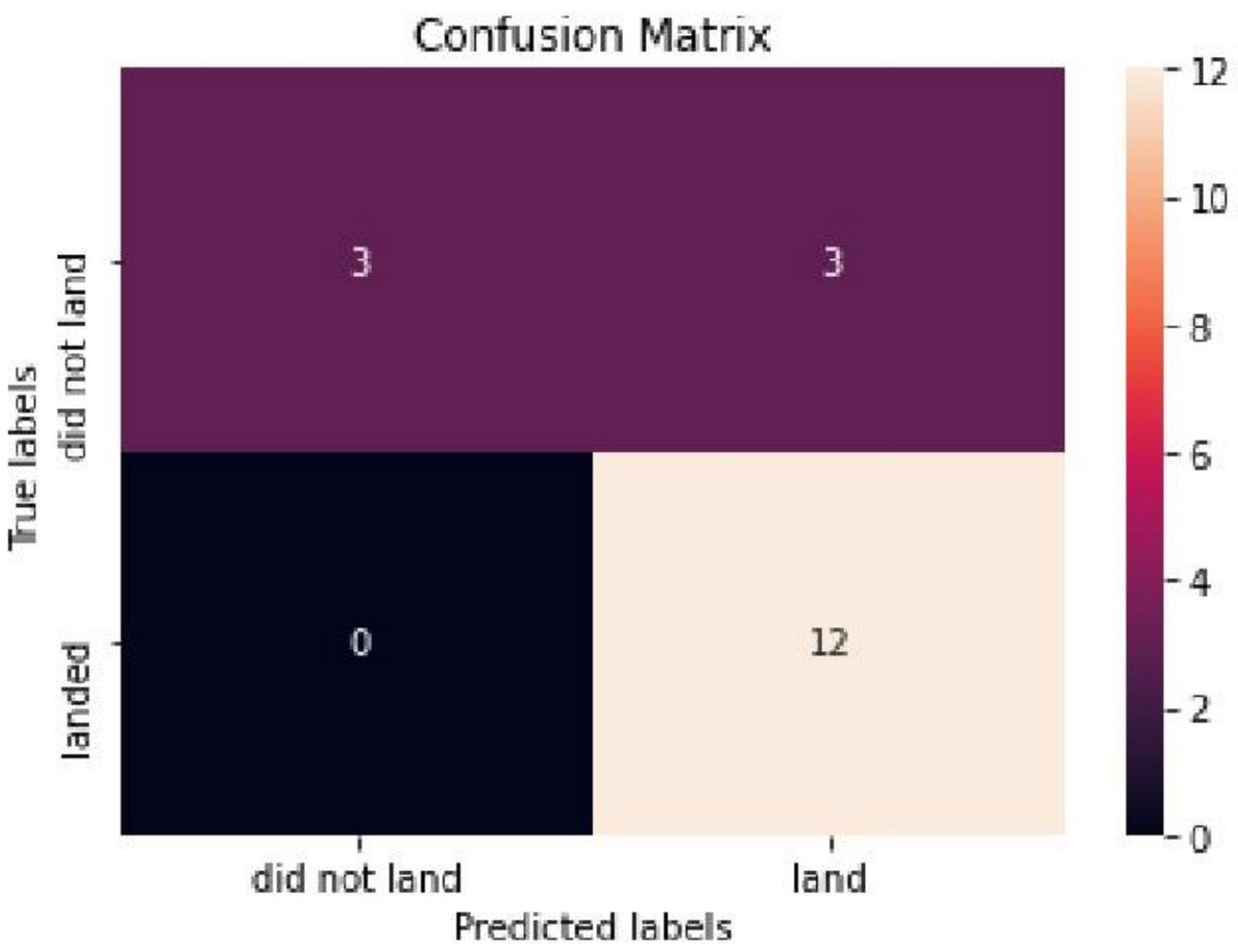
- The decision tree classifier demonstrated the highest classification accuracy among the models tested, making it the most effective for predicting landing outcomes.

```
models = {'KNeighbors':knn_cv.best_score_,
              'DecisionTree':tree_cv.best_score_,
              'LogisticRegression':logreg_cv.best_score_,
              'SupportVector': svm_cv.best_score_}

bestalgorithm = max(models, key=models.get)
print('Best model is', bestalgorithm,'with a score of', models[bestalgorithm])
if bestalgorithm == 'DecisionTree':
    print('Best params is :', tree_cv.best_params_)
if bestalgorithm == 'KNeighbors':
    print('Best params is :', knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best params is :', logreg_cv.best_params_)
if bestalgorithm == 'SupportVector':
    print('Best params is :', svm_cv.best_params_)
```

```
Best model is DecisionTree with a score of 0.8732142857142856
Best params is : {'criterion': 'gini', 'max_depth': 6, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_split': 5, 'splitter': 'random'}
```

Confusion Matrix

- The confusion matrix for the decision tree classifier indicates its ability to distinguish between classes effectively. However, a notable issue is the occurrence of false positives, where unsuccessful landings are incorrectly classified as successful. This highlights an area for further model refinement.



Conclusions

- A higher number of flights at a launch site is strongly associated with improved success rates.
- Launch success rates showed consistent growth from 2013 to 2020, reflecting operational advancements.
- Orbits such as ES-L1, GEO, HEO, SSO, and VLEO demonstrated the highest success rates.
- The KSC LC-39A launch site recorded the highest number of successful missions among all sites.
- The decision tree classifier proved to be the most effective machine learning model for this analysis.

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!