IBM Data Science Capstone Project

A ALI

OUTLINE

- Executive Summary
- Introduction
- Methodology
- Results
  - Visualization – Charts
  - Dashboard
- Discussion
  - Findings & Implications
- Conclusion
- Appendix

EXECUTIVE SUMMARY

- **Summary of methodologies**
  - Data Collection through API, Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis with SQL, Data Visualization
  - Interactive Visual Analytics with Folium
  - Machine Learning Prediction
- **Summary of all results**
  - Predictive Analytics and Exploratory Data Analysis result
  - Interactive visualizations of data analysis

INTRODUCTION

🚀 Project Background and Purpose

- SpaceX lists Falcon 9 rocket launches at $62 million, significantly lower than competitors who charge upwards of $165 million. This cost reduction is largely due to SpaceX's ability to reuse the rocket's first stage. By predicting whether the first stage will successfully land, we can estimate the launch cost. This insight is valuable for other companies aiming to compete with SpaceX in launch bids.
- The objective of this project is to develop a machine learning pipeline that can forecast the success of first-stage landings.

🔍 Questions to Investigate

- What key factors influence the rocket's landing success?
- How do different features interact to affect the likelihood of a successful landing?
- What operational conditions are necessary to consistently achieve successful landings?

METHODOLOGY

🚀 Executive Summary

- **Data Collection Approach**

The dataset was compiled using the SpaceX API and supplemented with information scraped from Wikipedia.

- **Data Preparation**

Categorical variables were transformed using one-hot encoding to make them suitable for machine learning models.

- **Exploratory Data Analysis (EDA)**

Initial insights were drawn using SQL queries and visualizations to understand feature distributions and relationships.

- **Interactive Visual Analytics**

Tools like Folium and Plotly Dash were used to create dynamic dashboards and geospatial visualizations for deeper exploration.

- **Predictive Modeling**

Classification algorithms were employed to predict the success of rocket landings.

- **Model Development Workflow**

The process included building, tuning, and evaluating classification models to optimize predictive performance.

DATA COLLECTION

📊 Data Collection and Preprocessing Overview

- To build a robust machine learning pipeline for predicting Falcon 9 first-stage landing success, we gathered and prepared data using multiple sources and techniques:

🚀 SpaceX API Integration

- We initiated data collection using HTTP GET requests to the official SpaceX API.
- The API response was decoded into JSON format using the .json() method.
- We then transformed the nested JSON structure into a flat tabular format using pandas.json_normalize().

🖌️ Data Cleaning

- The resulting dataframe was examined for missing values.
- Missing entries were handled appropriately through imputation or removal, ensuring data integrity for modeling.

🌐 Web Scraping from Wikipedia

- To supplement the API data, we scraped Falcon 9 launch records from Wikipedia using the BeautifulSoup library.
- The launch history was extracted from HTML tables, parsed, and converted into a pandas dataframe for further analysis.

1.DATA COLLECTION VIA SPACE X API

Data Acquisition and Preparation

- We initiated data collection using an HTTP GET request to the SpaceX API.
- The response was parsed into JSON format and converted into a structured pandas dataframe.
- Subsequently, we performed basic data cleaning, wrangling, and formatting to ensure consistency and usability for downstream analysis.

NOTEBOOK LINK BELOW:

- https://github.com/Aali715/DATA-SCIENCE-CAPSTONE-PROJECT/blob/9d1ee33c7e06ff05fc0e58f5fe9e420f2c2969c2/jupyter-labs-spacex-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
```

2.Data Collection Via Web Scraping

- **Web Scraping Falcon 9 Launch Records**
- **To enrich our dataset, we applied web scraping techniques to extract historical Falcon 9 launch records from Wikipedia using the `BeautifulSoup` library.**
- **We targeted the HTML table containing launch data, parsed its structure, and converted the extracted content into a structured `pandas` dataframe.**
- **This supplementary data provided valuable insights for our analysis and model training.**

LINK TO NOTEBOOK

- https://github.com/Aali715/DATA-SCIENCE-CAPSTONE-PROJECT/blob/9d1ee33c7e06ff05fc0e58f5fe9e420f2c2969c2/jupyter-labs-webscraping.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

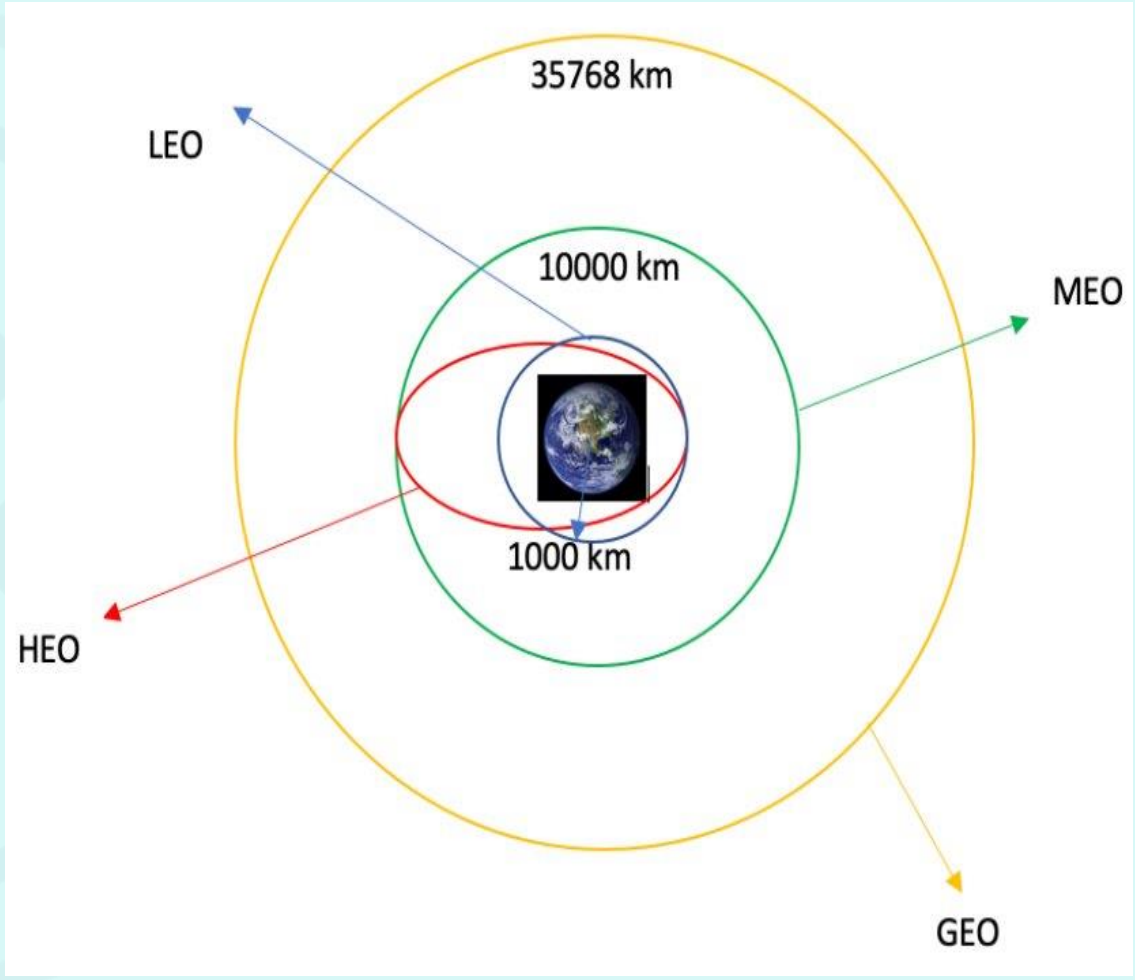
3.DATA WRANGLING

Exploratory Data Analysis and Label Preparation

- We conducted exploratory data analysis to understand the distribution and characteristics of the launch dataset.
- This included calculating the number of launches per site and analyzing the frequency and types of orbital paths.
- To prepare the target variable for model training, we derived a binary landing outcome label from the original outcome column.
- The processed results were then exported to a CSV file for further modeling.

LINK TO NOTEBOOK

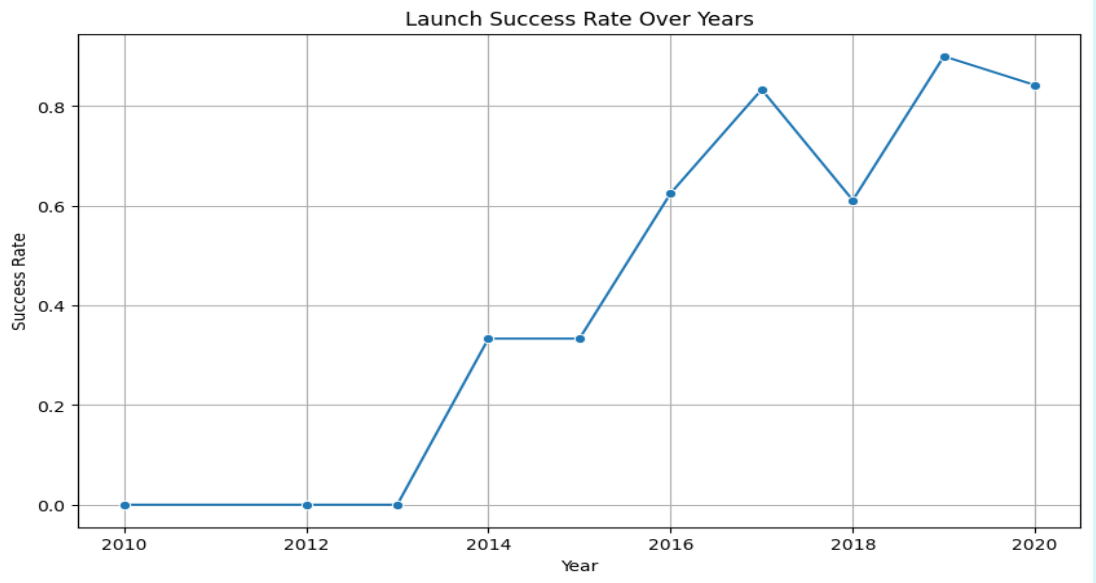
- https://github.com/Aali715/DATA-SCIENCE-CAPSTONE-PROJECT/blob/9d1ee33c7e06ff05fc0e58f5fe9e420f2c2969c2/labs-jupyter-spacex-Data%20wrangling.ipynb



EDA WITH DATA VISUALIZATION

1.EDA WITH DATA VISUALIZATION

- We explored the dataset by generating visualizations to uncover key relationships and trends using
- 1A Scatter,
- 1B Bar,
- 1C Line Graphs
- This included analyzing the correlation between flight number and launch site, payload mass and launch site, and the success rate across different orbit types.
- We also examined how flight number relates to orbit type and visualized the yearly trend of launch success to identify temporal patterns.



LINK TO NOTEBOOK
https://github.com/Aali715/DATA-SCIENCE-CAPSTONE-PROJECT/blob/9d1ee33c7e06ff05fc0e58f5fe9e420f2c2969c2/edadataviz.ipynb

1A SCATTER GRAPHS

Scatter Plot Overview: Exploring Attribute Dependencies

- To uncover patterns influencing launch success and landing outcomes, we visualized the following attribute pairs using scatter plots:
- **Payload vs. Flight Number**
- Assesses how payload mass trends evolve over time and across missions.
- **Flight Number vs. Launch Site**
- Reveals site-specific launch frequency and operational history.
- **Payload vs. Launch Site**
- Highlights payload capacity variations across different launch facilities.
- **Flight Number vs. Orbit Type**
- Tracks mission evolution across orbital categories over time.
- **Payload vs. Orbit Type**
- Investigates how payload mass correlates with target orbit classifications.

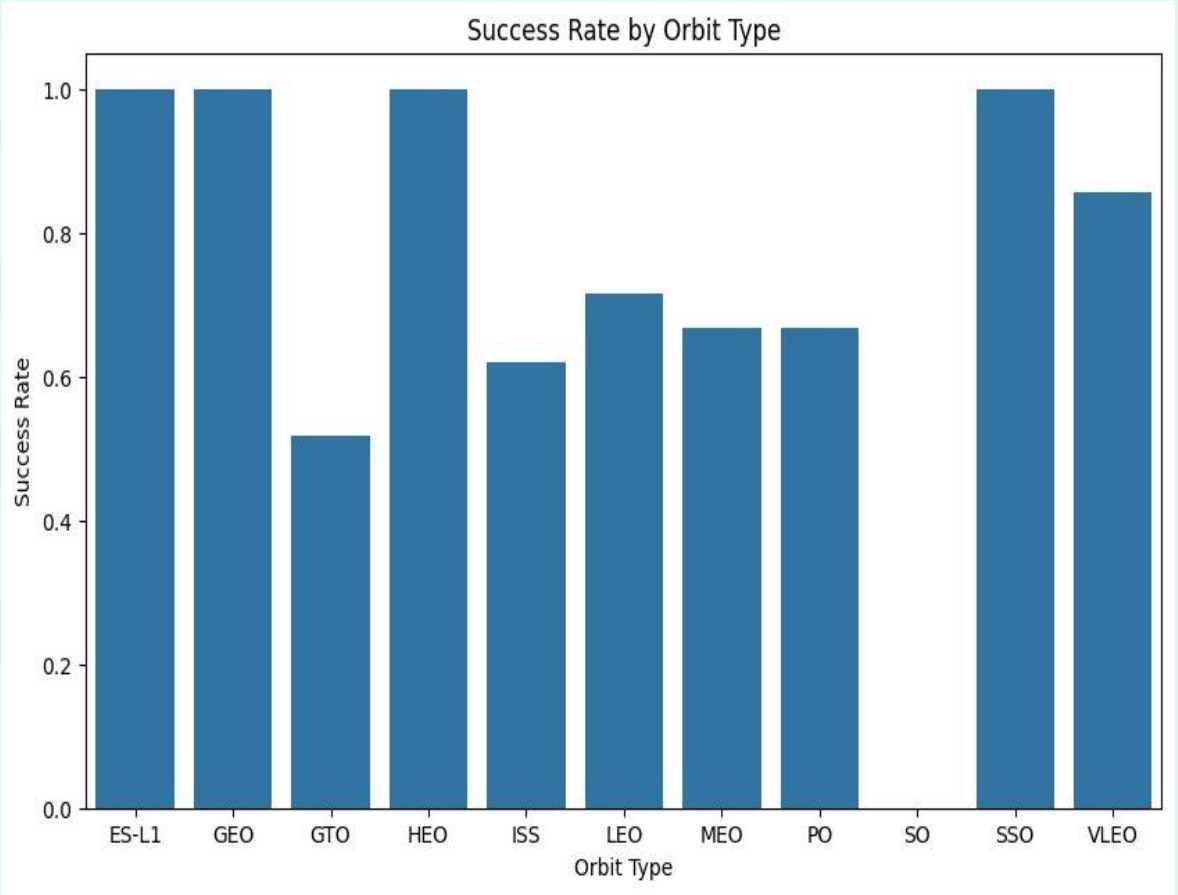
Scatter plots help identify attribute dependencies. Once patterns emerge, we can better predict which factors contribute to higher success probabilities for both mission outcomes and booster landings.

1B BAR GRAPH

📊 Bar Graph Insight: Success Rate by Orbit Type

- To evaluate how orbital destinations influence mission outcomes, we visualized **success rate versus orbit type** using a bar graph.
- **Purpose**: Identify which orbit types are most reliably associated with successful launches.
- **Interpretability**: Bar graphs offer a clear comparison across categories, making it easy to spot high-performing orbits.
- **Outcome**: This visualization helps pinpoint optimal orbital targets for maximizing launch success probability.

By comparing success rates across orbit types, we gain actionable insight into which missions are most likely to succeed — a key factor in planning future launches.

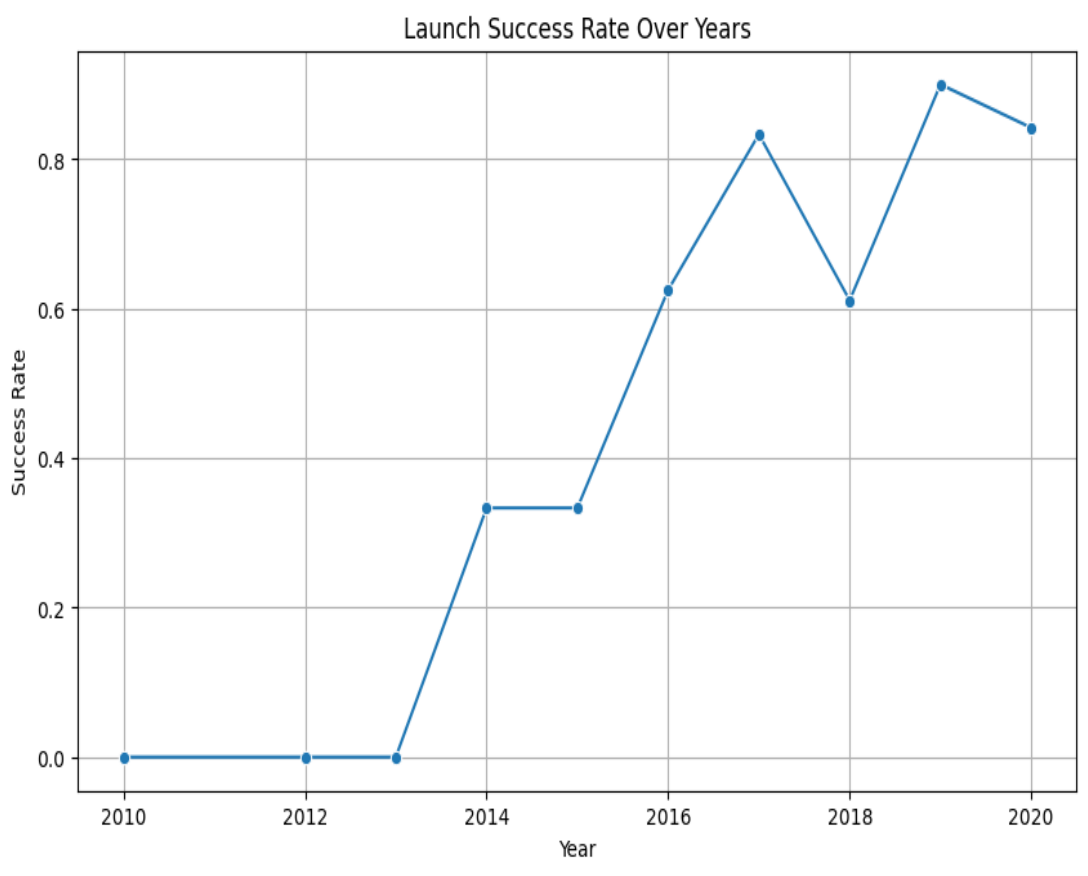


1C LINE GRAPH

- 📈 **Line Graph Insight: Yearly Launch Success Trend**

- To understand how SpaceX's launch reliability has evolved over time, we plotted **launch success rate by year** using a line graph.
- **Purpose**: Reveal long-term trends in mission success and assess operational improvements.
- **Strength**: Line graphs clearly show upward or downward trajectories, making them ideal for forecasting.
- **Outcome**: This visualization supports predictive modeling by highlighting consistent growth, volatility, or plateaus in success rates.

By tracking yearly performance, we gain strategic insight into SpaceX’s progress and can anticipate future reliability.



2.EDA WITH SQL

SQL-Based Analysis in PostgreSQL

- We seamlessly loaded the SpaceX dataset into a PostgreSQL database directly from the Jupyter Notebook environment.
- Using SQL queries, we performed exploratory data analysis to extract key insights, including:
- Identifying all unique launch sites involved in the missions
- Calculating the total payload mass carried by boosters launched under NASA's CRS program
- Determining the average payload mass for missions using the F9 v1.1 booster version
- Counting the total number of successful and failed mission outcomes
- Investigating failed drone ship landings, including associated booster versions and launch site names

LINK TO NOTEBOOK

https://github.com/Aali715/DATA-SCIENCE-CAPSTONE-PROJECT/blob/9d1ee33c7e06ff05fc0e58f5fe9e420f2c2969c2/jupyter-labs-eda-sql-coursera_sqllite.ipynb

3.INTERACTIVE VISUAL ANALYTICS

3A.BUILD AN INTERACTIVE MAP WTH FOLIUM
3B.BUILD A DASHBOARD WITH PLOTLYDASH

3A.BUILD AN INTERACTIVE MAP WTH FOLIUM

- **Geospatial Mapping and Proximity Analysis**
- We visualized all launch sites on a Folium map, enhancing it with interactive elements such as markers, circles, and lines to indicate the success or failure of each launch.
- Launch outcomes were encoded as binary classes — 0 for failure and 1 for success — to support classification and visualization.
- By leveraging color-coded marker clusters, we were able to visually assess which launch sites demonstrated higher success rates.
- We also computed distances from each launch site to nearby infrastructure and natural features to answer key spatial questions, such as:
- Are launch sites located near railways, highways, or coastlines?
- Do launch sites maintain a certain distance from urban areas?

LINK TO NOTEBOOK

- https://github.com/Aali715/DATA-SCIENCE-CAPSTONE-PROJECT/blob/9d1ee33c7e06ff05fc0e58f5fe9e420f2c2969c2/lab_jupyter_launch_site_location.ipynb

3B.BUILD A DASHBOARD WITH PLOTLY DASH

- 📊 **Interactive Dashboard with Plotly Dash**
- We developed an interactive dashboard using Plotly Dash to visualize key insights from the launch dataset.
- The dashboard includes:
- **Pie charts** illustrating the distribution of total launches across various launch sites.
- **Scatter plots** depicting the relationship between launch outcomes and payload mass (in kilograms), segmented by booster version categories.
- These visualizations enable dynamic exploration of launch performance and payload characteristics across different configurations.

LINK TO NOTEBOOK

https://github.com/Aali715/DATA-SCIENCE-CAPSTONE-PROJECT/blob/35cec9b1b0ecc4fa9a7db36c79c33ac0c398733e/app.py

Predictive Analysis (Classification)

🚀 Machine Learning Pipeline Summary

1. Data Loading & Preparation

- Imported data using **NumPy** and **Pandas**.
- Performed **data transformation** to clean and structure the dataset.
- Split the dataset into **training and testing sets** for model evaluation.

2. Model Building & Tuning

- Trained multiple **classification models** (e.g., Logistic Regression, Random Forest, SVM, etc.).
- Used **GridSearchCV** to perform **hyperparameter tuning**, ensuring optimal model configurations.

3. Evaluation & Optimization

- Chose **accuracy** as the primary evaluation metric.
- Applied **feature engineering** techniques to enhance model performance.
- Iteratively improved results through **algorithm tuning**.

4. Model Selection

- Identified the **best-performing classification model** based on validation accuracy.

LINK TO NOTEBOOK
https://github.com/Aali715/DATA-SCIENCE-CAPSTONE-PROJECT/blob/250953ca1333d52c0b1b10ac90a8d10882697c7d/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

IBM

RESULTS

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

EDA WITH VISUALIZATION

1. Flight Number vs. Launch site scatter chart
2. Payload vs. Launch site scatter chart
3. Success rate vs. Orbit type bar chart
4. Flight Number vs. Orbit type scatter chart
5. Payload vs. Orbit type scatter chart
6. Launch success yearly trend line

LINK TO NOTE BOOK

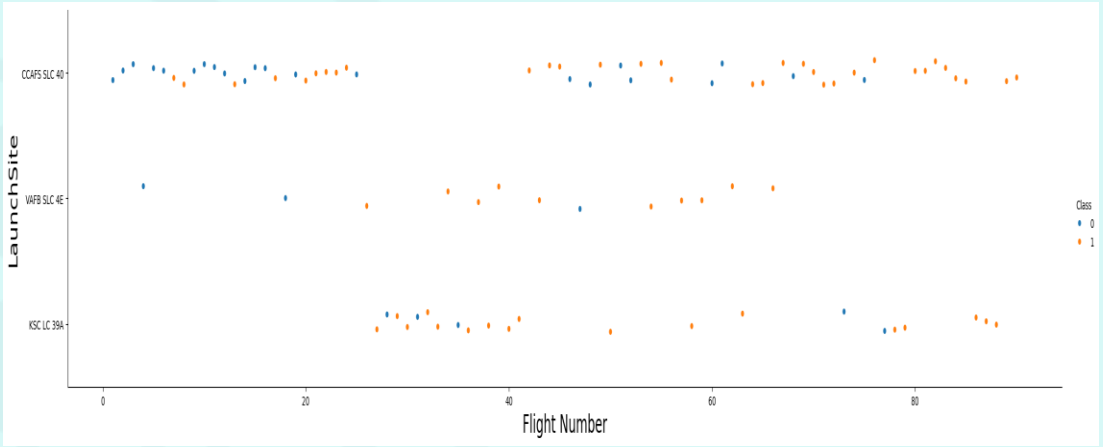
- EDA WTH VISUALIZATION
- https://github.com/Aali715/DATA-SCIENCE-CAPSTONE-PROJECT/blob/35cec9b1b0ecc4fa9a7db36c79c33ac0c398733e/edadataviz.ipynb

1.Flight Number vs. Launch Site Scatter Chart

🚀 Launch Frequency vs. Success Rate

From the plotted data, we observed a positive correlation between the number of flights at a launch site and its success rate.

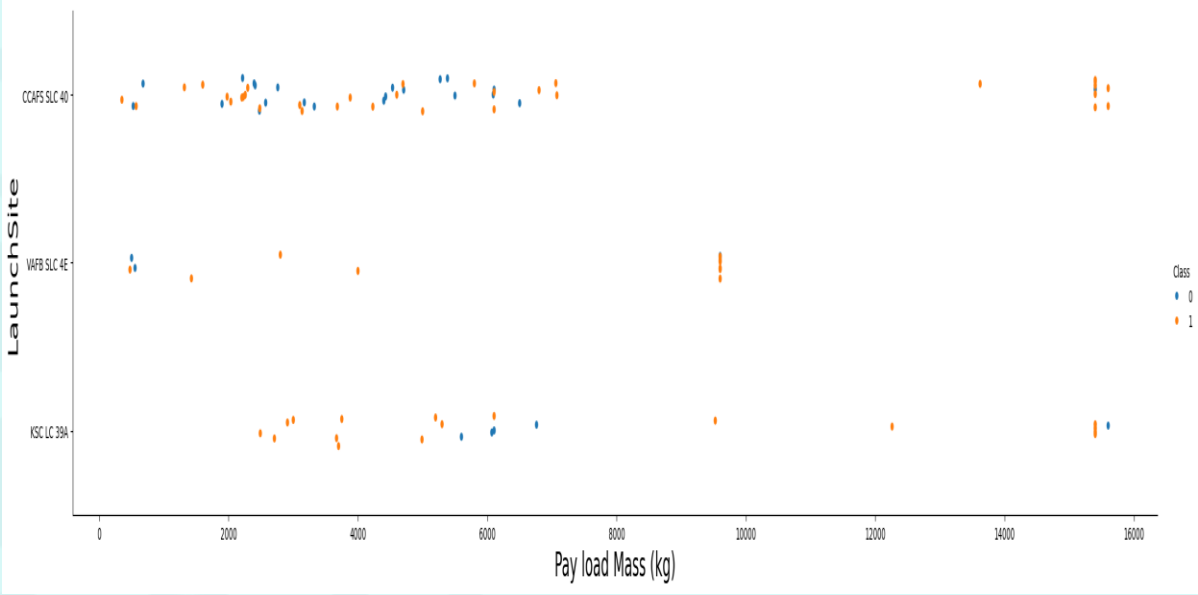
https://github.com/Aali715/DATA-SCIENCE-CAPSTONE-PROJECT/blob/35cec9b1b0ecc4fa9a7db36c79c33ac0c398733e/edadataviz.ipynb



2.PAYLOAD VERSUS LAUNCH SITE SCATTER CHART

Payload Mass vs. Success Rate at CCAFS SLC 40

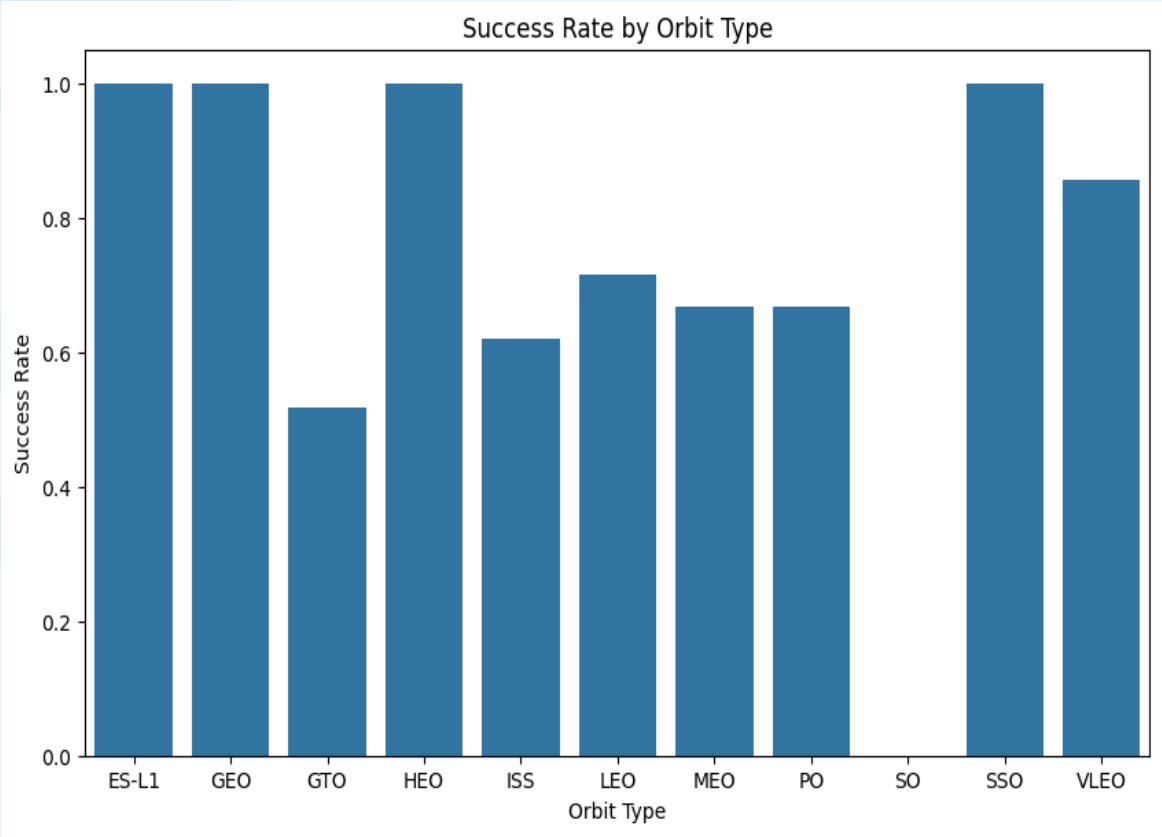
- At Cape Canaveral's SLC 40, rockets carrying heavier payloads are associated with higher success rates. This may reflect better mission planning, more robust rocket configurations, or selective launch conditions for heavier payloads.



3.SUCCESS RATE VERSUS ORBIT TYPE BAR CHART

Orbit Type vs. Launch Success Rate

- From the plotted data, we observed that the following orbit types — **ES-L1, GEO, HEO, SSO, and VLEO** — exhibited the highest success rates.



4.FLIGHT NUMBER vs. ORBIT TYPE SCATTER CHART

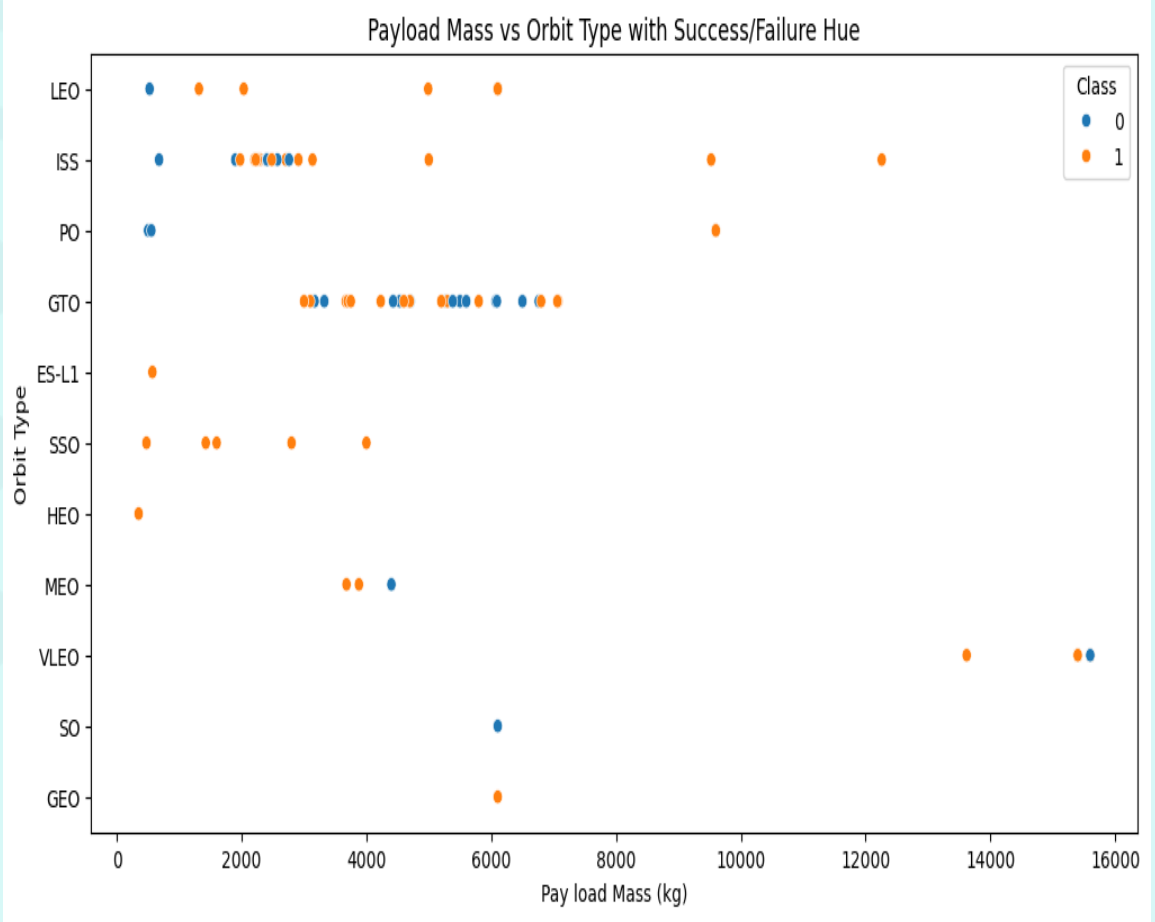
Flight Number vs. Orbit Type

- The plot reveals a notable trend in **LEO (Low Earth Orbit)** missions:
- **Higher flight numbers** are generally associated with **increased success rates**, suggesting that operational experience and iterative improvements may enhance reliability for LEO launches.
- In contrast, for **GTO (Geostationary Transfer Orbit)** missions:
- There appears to be **no clear correlation** between flight number and success, indicating that other factors — such as mission complexity, payload characteristics, or external conditions — may play a more dominant role in determining outcomes.

5.PAYLOAD vs. ORBIT TYPE SCATTER CHART

Payload Mass vs. Orbit Type and Landing Success

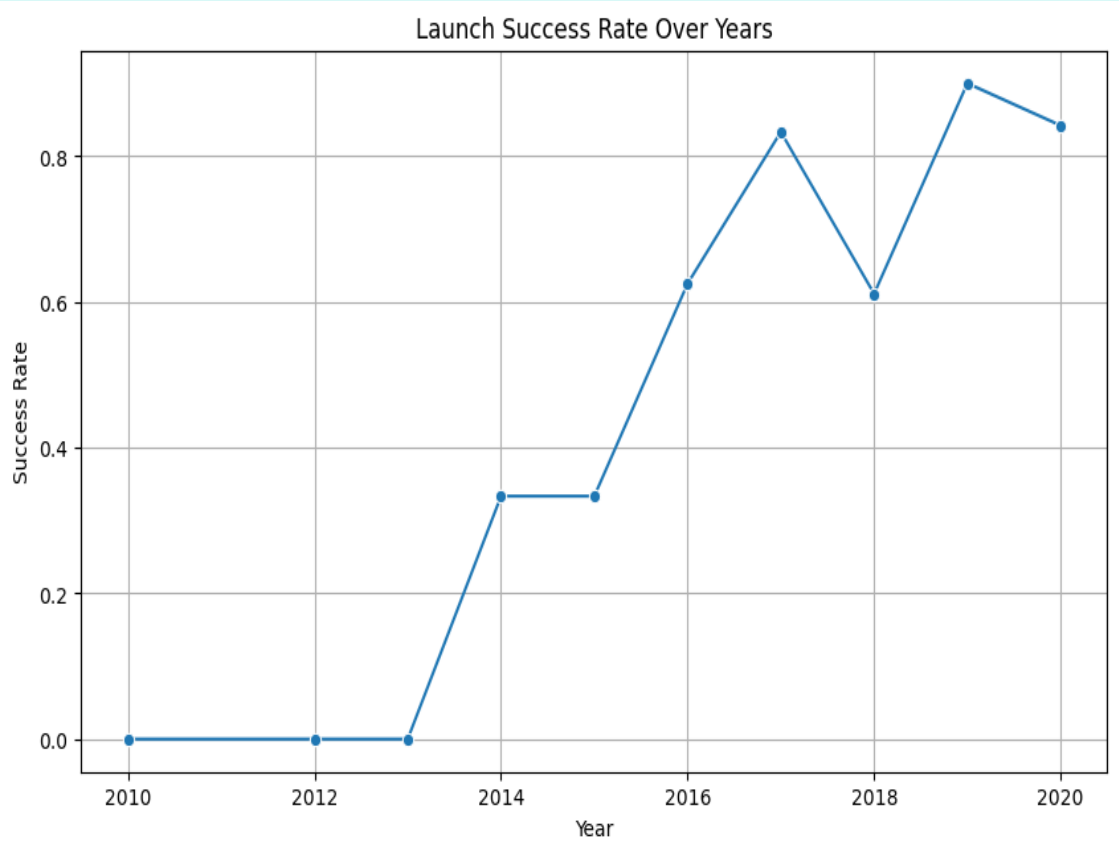
- The data suggests that **heavier payloads** are more frequently associated with **successful landings** when targeting the following orbits:
- **PO (Polar Orbit)**
- **LEO (Low Earth Orbit)**
- **ISS (International Space Station)**
- **This trend may reflect the operational maturity and mission familiarity of these orbits, where robust planning, optimized rocket configurations, and refined landing protocols contribute to higher success rates — even under heavier payload conditions.**



6.LAUNCH SUCCESS YEARLY TREND LINE

Temporal Trend in Launch Success (2013–2020)

- The plotted data reveals a **steady upward trend in launch success rates** from **2013 through 2020**.
- This consistent improvement likely reflects SpaceX's growing operational expertise, iterative engineering enhancements, and increased reliability of booster recovery systems over time.



EXOPLORATORY DATA ANALYSIS WITH SQL

LINK TO NOTEBOOK

https://github.com/Aali715/DATA-SCIENCE-CAPSTONE-PROJECT/blob/1c9e18821454b414f2a25913c7375013e857e5be/jupyter-labs-eda-sql-coursera_sqllite.ipynb

10 EDA with SQL related slides/queries:

1. All launch site names: Find the names of the unique sites
2. Launch site names begin with `CCA`: Find all launch site names begin with `CCA`
3. Total payload mass: Calculate the total payload carried by booster from NASA
4. Average payload mass by F9 v1.1: Calculate the average payload mass carried by booster F9 v1.1
5. First successful ground landing date: find the date when the first successful landing outcome in ground pad
6. Successful drone ship landing with payload between 4000 and 6000: List the names of boosters which have success in drone ship and have payload greater than 4000 but less than 6000
7. Total number of successful and failure mission outcomes: Calculate the total number of successful and failure mission outcomes
8. Boosters carried maximum payload: List the names of the booster which have carried the maximum payload mass
9. 2015 launch records: List the launch records for months in 2015
10. Rank success count between 2010-06-04 and 2017-03-20: Rank the count of successful landings between 2010-06-04 and 2017-03-20

1.ALL LAUNCH SITE NAMES

Extracting Unique Launch Sites with SQL

- To identify **distinct launch sites** from the SpaceX dataset, we used the SQL keyword DISTINCT. This ensures that each launch site appears only once in the query result, eliminating duplicate entries.

Task 1

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

In [41]:
```
%sql SELECT DISTINCT launch_site FROM SPACEXTBL;
```

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

Out[41]:

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

2.LAUNCH SITE NAMES BEGIN WITH 'CCA'

🔍 Filtering Launch Sites by Prefix

- To retrieve a sample of launch sites that **begin with 'CCA'**, we extended our previous query using the LIKE operator with a wildcard and limited the output to just 5 records for inspection.
- **LIKE 'CCA%'** filters for launch site names that start with 'CCA'.
- **LIMIT 5** restricts the output to the first five matching records.
- This query helps quickly verify the presence and format of launch sites associated with Cape Canaveral, such as 'CCAFS LC-40'.

Task 2

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

In [42]:
```
%sql SELECT * FROM SPACEXTBL WHERE launch_site LIKE 'CCA%' LIMIT 5;
```

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

Out[42]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

3.TOTAL PAYLOAD MASS: Total payload carried by booster from NASA

The SQL query returned a cumulative payload of 45,596 kg for NASA-contracted booster missions.

- **WHERE Customer = 'NASA (CRS)'** → filters the dataset to only NASA missions.
- **SUM(PayloadMass)** → adds up all payload masses for those missions.
- **AS Total_Pay_load** → labels the result column for clarity.

Task 3

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

In [44]:
```
%sql SELECT SUM(payload_mass__kg_) AS total_payload_mass FROM SPACEXTBL WHERE customer = 'NASA (CRS)';
```

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

Out[44]:

total_payload_mass
45596

4.AVERAGE PAYLOAD MASS BY F9 v1.1

An aggregate SQL query was executed to determine the mean payload mass associated with Falcon 9 v1.1 booster missions. The computed average payload was approximately 2,928.4 kilograms.

- **Method** → aggregate SQL query
- **Scope** → Falcon 9 v1.1 booster missions
- **Result** → 2,928.4 kg

Task 4

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

In [46]:
```
%sql SELECT AVG(payload_mass__kg_) AS average_payload_mass FROM SPACEXTBL WHERE booster_version = 'F9 v1.1';
```

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

Out[46]:

average_payload_mass
2928.4

5. FIRST SUCCESS GROUND LANDING DATE

The date for the first successful landing outcome in ground pad:

Our observation shows that the first ground pad landing success took place on December 22, 2015.

Task 5

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

Hint:Use min function

In [47]:
```
%sql SELECT MIN(date) AS first_successful_landing_date FROM SPACEXTBL WHERE landing_outcome = 'Success (ground pad)';
```

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

Out[47]:

first_successful_landing_date
2015-12-22

6.Successful Drone Ship Landing with Payload between4000 and 6000

A WHERE clause was applied to isolate boosters that achieved successful drone-ship landings. An AND condition was then introduced to further constrain the results to missions with payload masses between 4,000 and 6,000 kilograms.

- **Method** → filtering with WHERE
- **Criteria** → success on drone ship
- **Constraint** → payload mass range

Task 6

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

In [48]:
```
%sql SELECT DISTINCT booster_version FROM SPACEXTBL WHERE landing_outcome = 'Success (drone ship)' AND payload_mass__kg_ > 4
```

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

Out[48]:

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

7.TOTAL NUMBER OF SUCCESSFUL AND FAILURE MISSION OUTCOMES

A wildcard operator (%) was applied within the WHERE clause to filter records where the MissionOutcome indicated either success or failure.

Task 7

List the total number of successful and failure mission outcomes

```
[21]: %sql SELECT "Mission_Outcome", COUNT(*) AS total_count FROM SPACEXTBL GROUP BY "Mission_Outcome";
```

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

[21]:

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

8.BOOSTERS CARRIERS MAXIMUM PAYLOAD

- We identified the booster with the highest payload by combining a subquery in the WHERE clause with the MAX() function.

Task 8

List all the booster_versions that have carried the maximum payload mass, using a subquery with a suitable aggregate function.

[22]:
```
%sql SELECT booster_version FROM SPACEXTBL WHERE payload_mass__kg_ = (SELECT MAX(payload_mass__kg_) FROM SPACEXTBL);
```

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

[22]:

Booster_Version
F9 B5 B1048.4
F9 B5 B1049.4
F9 B5 B1051.3
F9 B5 B1056.4
F9 B5 B1048.5
F9 B5 B1051.4
F9 B5 B1049.5
F9 B5 B1060.2
F9 B5 B1058.3
F9 B5 B1051.6
F9 B5 B1060.3
F9 B5 B1049.7

9.2015 LAUNCH RECORDS

- A combination of WHERE, LIKE, AND, and BETWEEN conditions was employed to filter records of failed drone-ship landings, along with their corresponding booster versions and launch site names, specifically for the year 2015.

Task 9

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

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

```
[42]: come",booster_version,launch_site FROM SPACEXTBL WHERE substr(Date, 0, 5) = '2015' AND "landing_outcome"='Failure (drone ship)';
```

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

[42]:

month	Landing_Outcome	Booster_Version	Launch_Site
01	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

10.Rank Landing Outcomes Between 2010-06-04 and2017-03-20

- We retrieved landing outcomes along with their counts, applying a WHERE clause to filter results within the date range from June 4, 2010 to March 20, 2017.
- The GROUP BY clause was then used to categorize the landing outcomes, and the ORDER BY clause arranged these grouped results in descending order.

Task 10

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

```
[45]: %%sql SELECT "landing_outcome",COUNT(*) AS outcome_count FROM SPACEXTBL WHERE Date BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY "landing_outcome" ORDER BY outcome_count DESC;
```

Done.

[45]:

Landing_Outcome	outcome_count
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

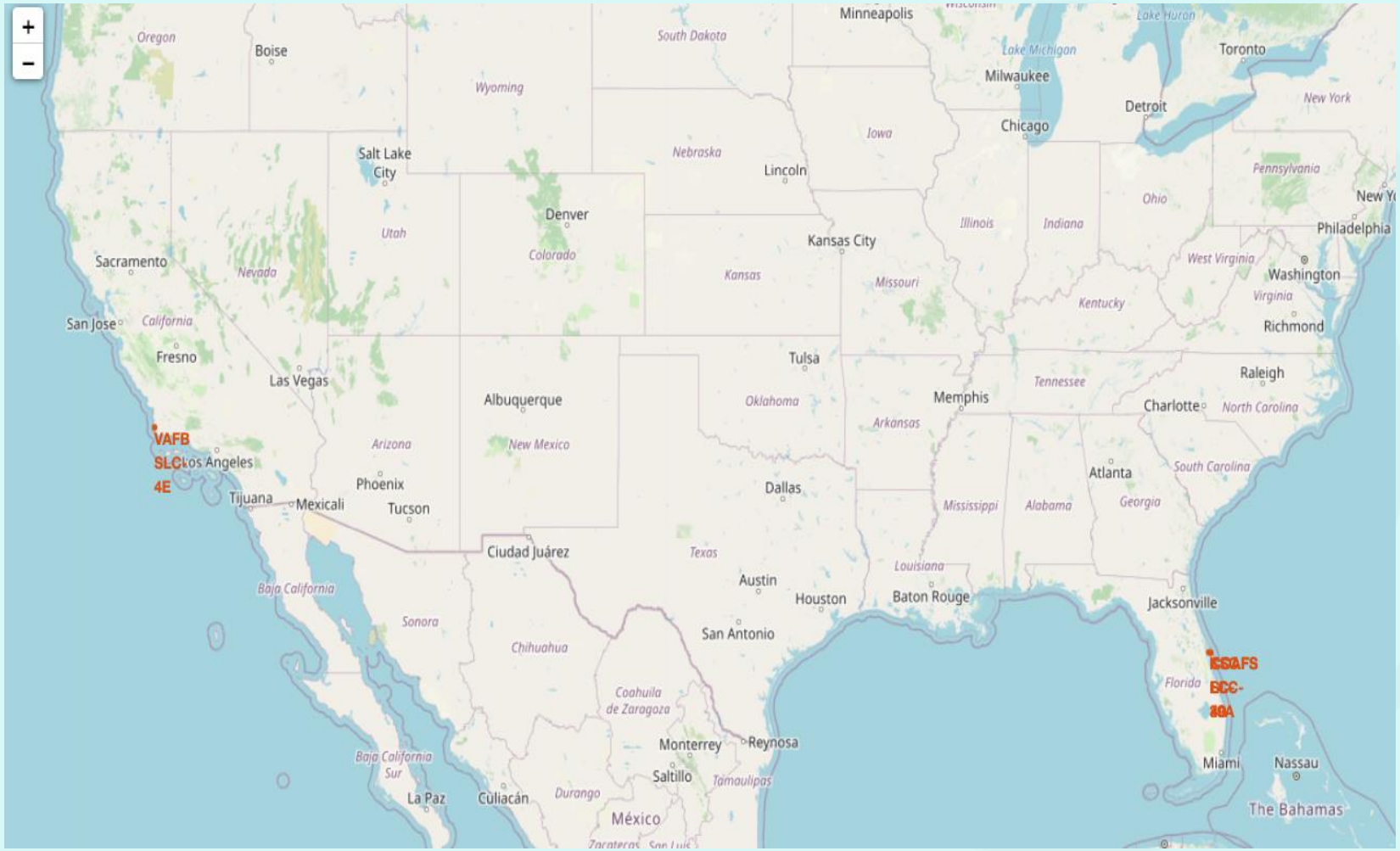
INTERACTIVE VISUAL ANALYTICS WITH FOLIUM

LINK TO NOTEBOOK

- https://github.com/Aali715/DATA-SCIENCE-CAPSTONE-PROJECT/blob/9d1ee33c7e06ff05fc0e58f5fe9e420f2c2969c2/lab_jupyter_launch_site_location.ipynb

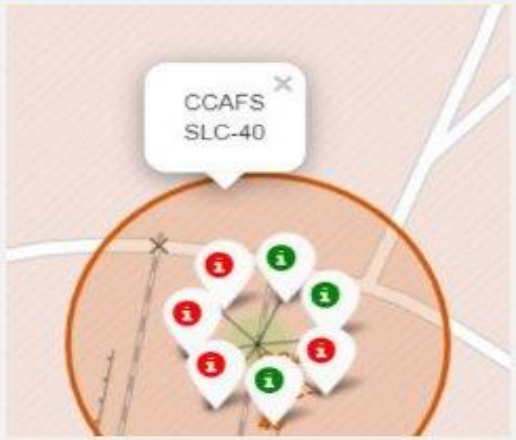
1. Screenshot and explanations of all launch sites' markers on a global map
2. Screenshot and explanations of all launch records per site on the map
3. Screenshot and explanations of launch sites' proximities such as railway, highway, coastline, with distance calculated and displayed

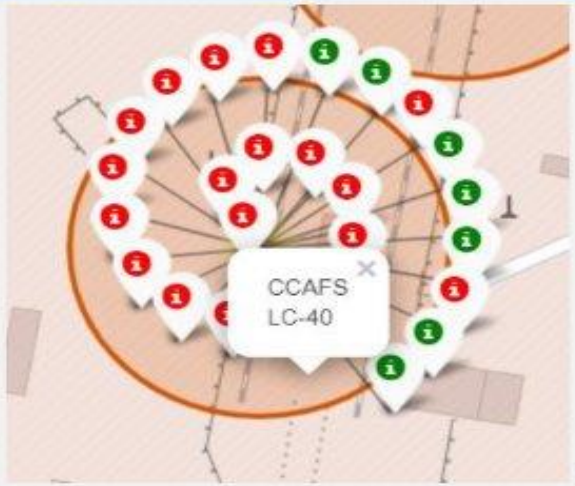
1.ALL LAUNCH SITES MARKERS ON A GLOBAL MAP

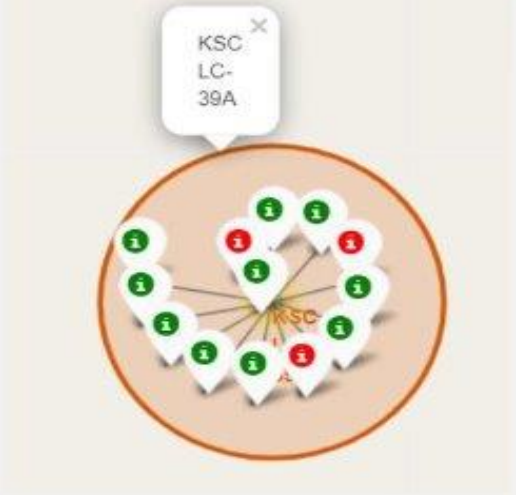


- The launch sites are strategically located along the U.S. coasts to optimize orbital trajectories and ensure safe recovery zones.
- Florida and California

2.MARKERS SHOWING LAUNCH SITES WITH COLOR LABELS





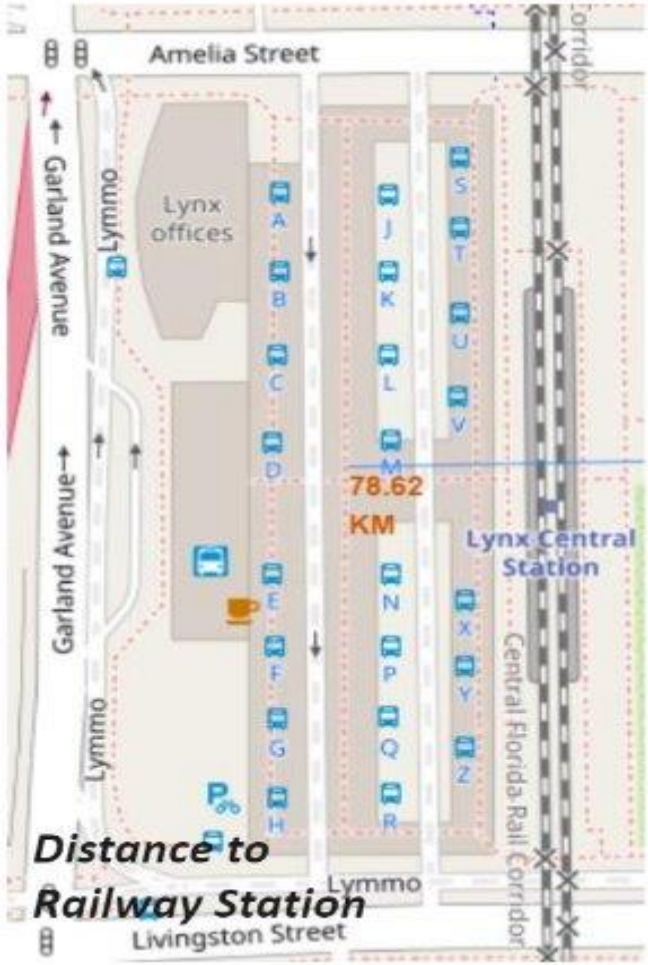


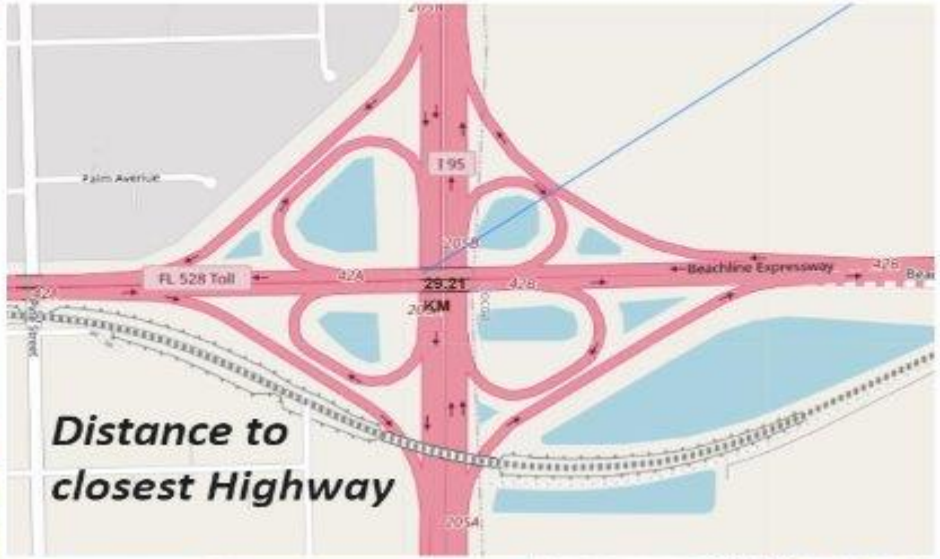
Florida Launch Sites

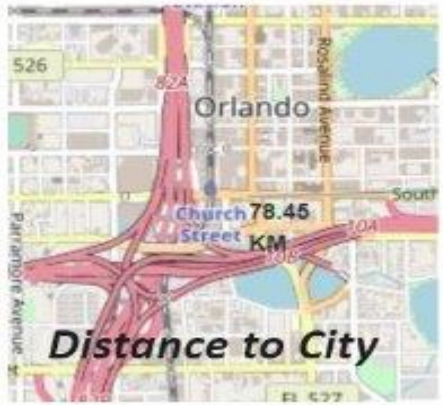
Green Marker shows successful Launches and Red Marker shows Failures

California Launch Site

3.LAUNCH SITE DISTANCE TO LANDMARKS

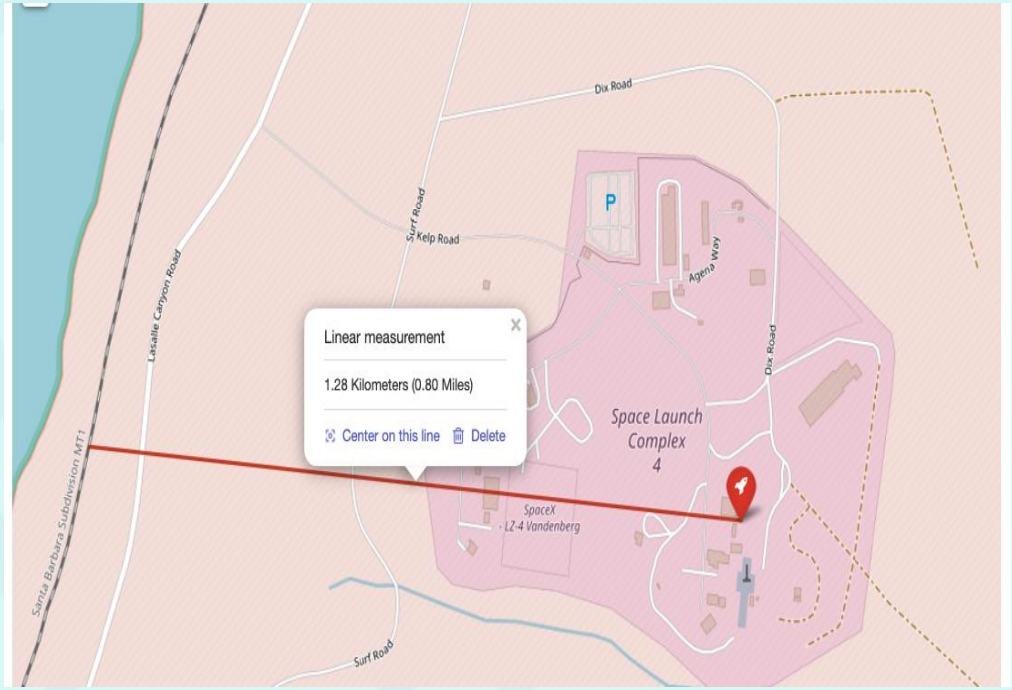






- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

3a.LAUNCH SITE DISTANCES FROM EQUATOR AND RAILWAYS



🌎 **Geospatial Context: Launch Site Latitude Constraint**

All SpaceX launch sites used in this analysis are located more than 3,000 km from the Equator.

Launch sites are more than 0.7 km from railway tracks.It suggests the sites are close enough for convenient transport access but far enough to avoid safety or vibration concerns from passing trains.

3b.LAUNCH SITE DISTANCE FROM COAST LINE AND CITY

All launch sites are located within four kilometers of the coastline.

If every launch site is more than 14 km from a city, we can conclude they're all far away.

3c. LAUNCH SITE DISTANCE FROM HIGHWAYS

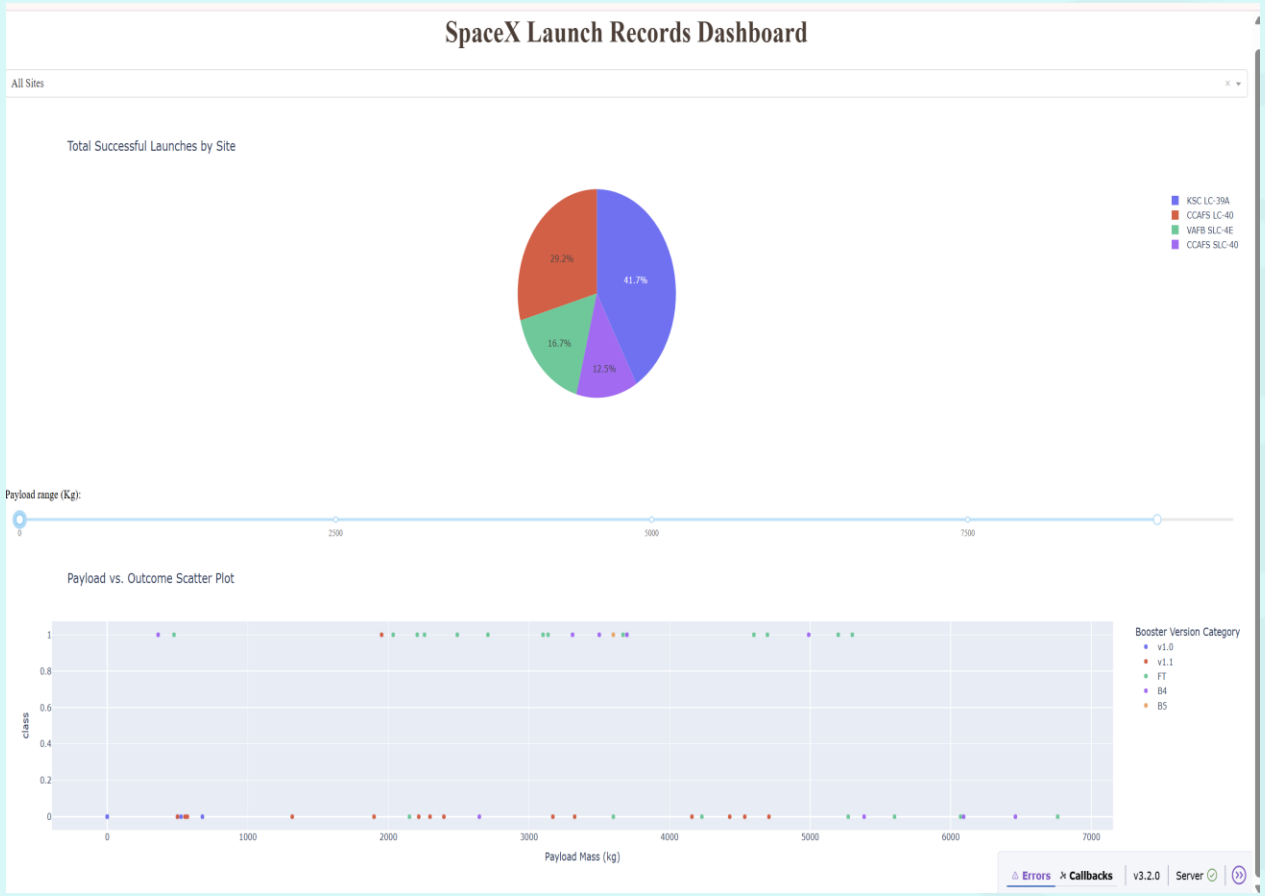
All launch sites are situated at distances exceeding 5 kilometers from the nearest highways, indicating a consistent spatial separation between launch infrastructure and major roadways.

BUILD A DASHBOARD WITH PLOTLY DASH

LINK TO NOTEBOOK

- https://github.com/Aali715/DATA-SCIENCE-CAPSTONE-PROJECT/blob/9d1ee33c7e06ff05fc0e58f5fe9e420f2c2969c2/lab_jupyter_launch_site_location.ipynb

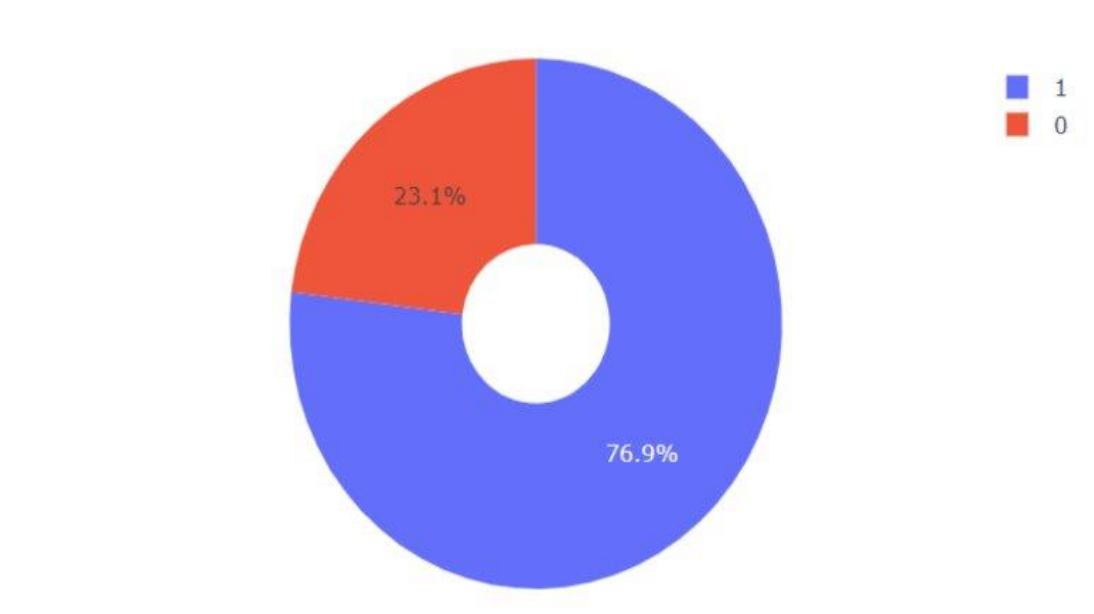
1.PIE CHART SHOWING THE SUCCESS OF EACH LAUNCH SITE IN %



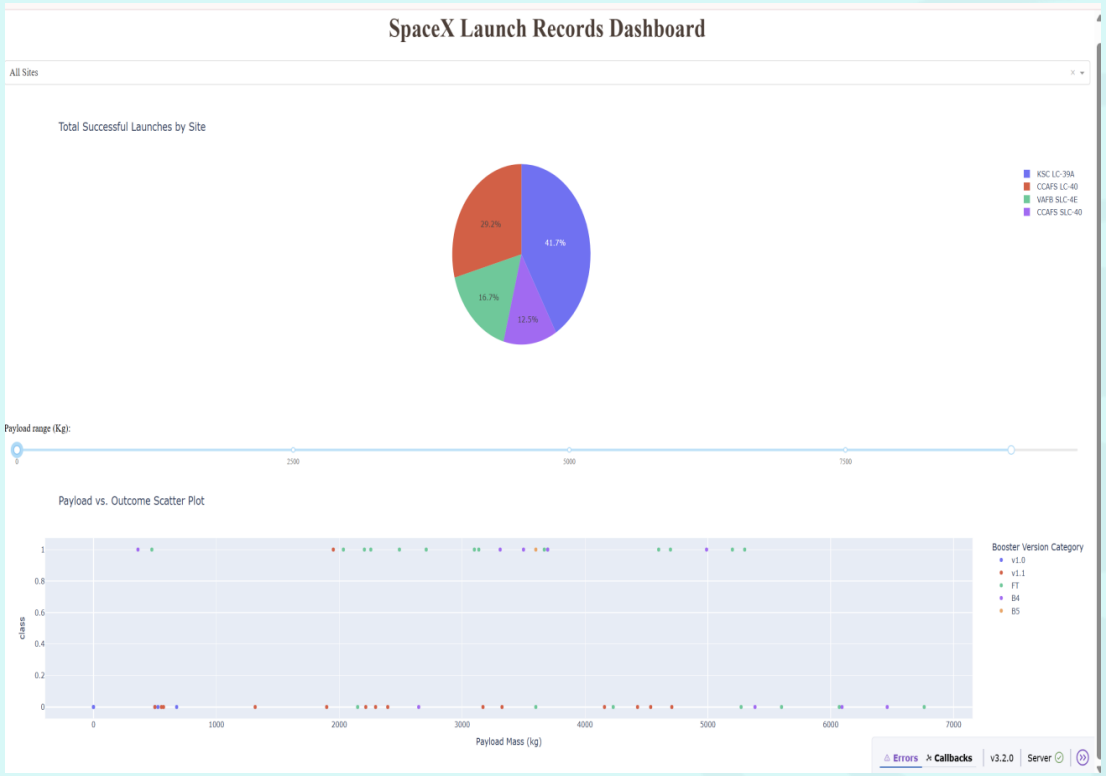
- **Success Rate by Site: KSC LC-39A leads with 41.7% of total successful launches, followed by CCAFS LC-40 (29.2%), VAFB SLC-4E (16.7%), and CCAFS SLC-40 (12.5%).**

2.PIE CHART DEPICTING THE LAUNCH SITE WITH THE HIGHEST LAUNCH SUCCESS RATIO

- KSC LC-39A has a 76.9% success rate and a 23.1% failure rate



3.SCATTER PLOT OF PAYLOAD vs LAUNCH OUTCOME FOR ALL SITES, PAYLOAD IN RANGE SLIDER



🚀 Payload Mass vs. Launch Success Rate

- To better understand how payload mass influences launch outcomes, we split the data into two categories:

1. **Low Weighted Payloads (0–4000 kg)**

- **Observation**: Higher concentration of successful launches.
- **Insight**: Missions with lighter payloads tend to have better reliability, possibly due to reduced mechanical stress and simpler launch configurations.

2. **Heavy Weighted Payloads (4000–10000 kg)**

- **Observation**: More dispersed success rates with visible failures.
- **Insight**: Heavier payloads may introduce greater complexity, increasing the likelihood of mission failure.
- 📊 *Conclusion*: Launch success rates are notably higher for lighter payloads, suggesting payload mass is a critical factor in mission planning and risk assessment.

PREDICTIVE ANALYSIS(CLASSIFICATION)

LINK TO NOTEBOOK
https://github.com/Aali715/DATA-SCIENCE-CAPSTONE-PROJECT/blob/250953ca1333d52c0b1b10ac90a8d10882697c7d/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

1.KNN MODEL PERFORMANCE

TASK 11

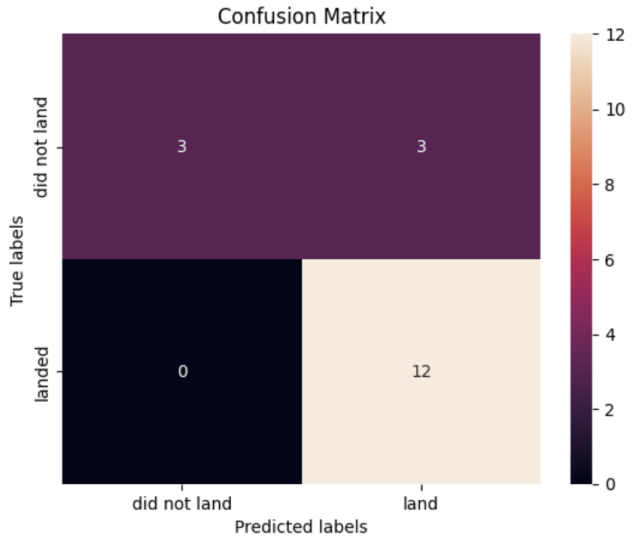
Calculate the accuracy of knn_cv on the test data using the method `score`:

In [39]:
```
test_accuracy = knn_cv.score(X_test, Y_test)
print("Test accuracy of knn_cv:", test_accuracy)
```

```
Test accuracy of knn_cv: 0.8333333333333334
```

We can plot the confusion matrix

In [40]:
```
yhat = knn_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



K-Nearest Neighbors (KNN) Model Performance Summary

- **Test Accuracy**: The KNN classifier achieved an accuracy of **83.33%** on the test dataset.
- **Confusion Matrix Insights**:
- Correctly predicted **12** successful landings.
- Correctly predicted **3** failed landings.
- Misclassified **3** failed landings as successful.
- No false negatives (i.e., no successful landings predicted as failures).

This indicates the model is highly effective at identifying successful launches, though it struggles slightly with distinguishing failed ones.

2.Logistic regression model performance

TASK 5

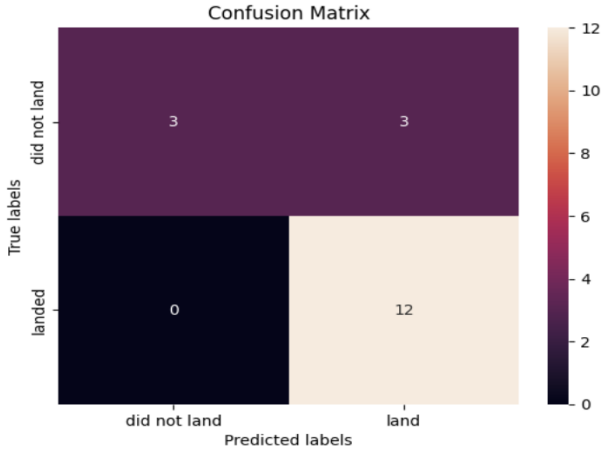
Calculate the accuracy on the test data using the method `score`:

In [15]:
```
# Calculate accuracy on test data
test_accuracy = logreg_cv.score(X_test, Y_test)
print("Test accuracy:", test_accuracy)
```

```
Test accuracy: 0.8333333333333334
```

Lets look at the confusion matrix:

In [16]:
```
yhat=logreg_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



Examining the confusion matrix, we see that logistic regression can distinguish between the different classes. We see that the problem is false positives.

Overview:

True Postive - 12 (True label is landed, Predicted label is also landed)

False Postive - 3 (True label is not landed, Predicted label is landed)

Logistic Regression Model Performance Summary

- **Test Accuracy**: The model achieved an accuracy of **83.33%** on the test dataset.
- **Confusion Matrix Insights**:
- Correctly predicted **12** successful landings.
- Correctly predicted **3** failed landings.
- Misclassified **3** successful landings as failures.
- No false positives (i.e., no failed landings predicted as successful).

This suggests the model is strong at identifying failed launches but occasionally misclassifies successful ones, highlighting a tendency toward false negatives.

3.Decision tree model performance

TASK 9

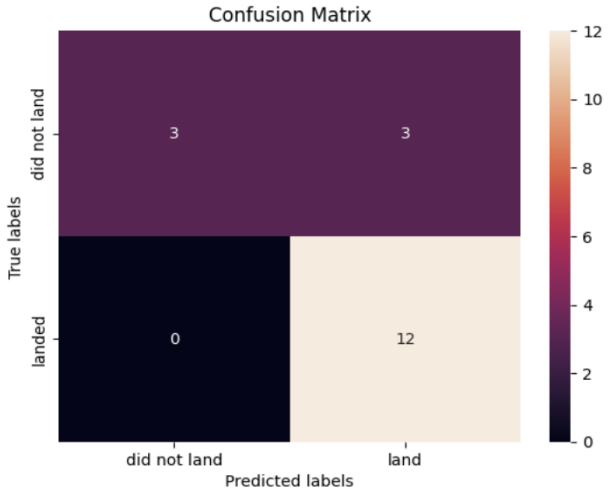
Calculate the accuracy of tree_cv on the test data using the method `score`:

In [30]:
```
accuracy = tree_cv.score(X_test, Y_test)
print("Test accuracy of tree_cv:", accuracy)
```

```
Test accuracy of tree_cv: 0.8333333333333334
```

We can plot the confusion matrix

In [31]:
```
yhat = tree_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



- **Decision Tree Model Performance Summary**
- **Test Accuracy**: The model achieved an accuracy of **83.33%**, matching the logistic regression model.
- **Confusion Matrix Breakdown**:
- **True Positives (Landed correctly predicted)**: 12
- **True Negatives (Did not land correctly predicted)**: 3
- **False Positives (Predicted landed but did not land)**: 3
- **False Negatives (Predicted did not land but actually landed)**: 0

This model tends to overpredict successful landings, resulting in **false positives**, but it never misses an actual landing—no false negatives.

4.SVM model performance

TASK 7

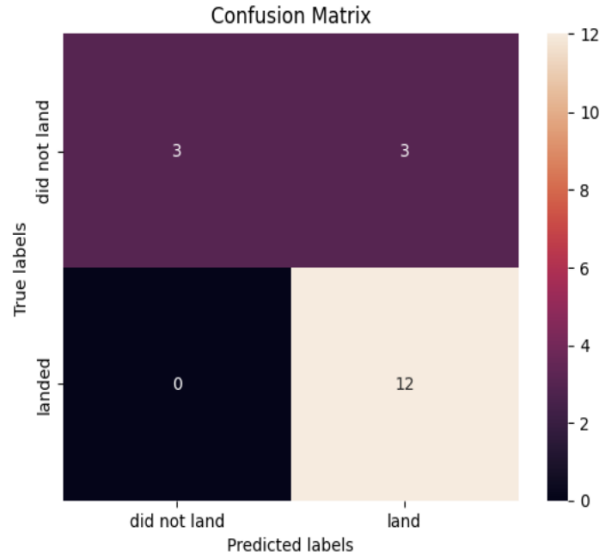
Calculate the accuracy on the test data using the method `score`:

In [21]:
```
test_accuracy = svm_cv.score(X_test, Y_test)
print("Test accuracy:", test_accuracy)
```

```
Test accuracy: 0.8333333333333334
```

We can plot the confusion matrix

In [22]:
```
yhat=svm_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



- ✅ **SVM Model Performance Summary**
- **Model Used:** Support Vector Machine (SVM) with cross-validation (svm_cv)
- **Test Accuracy:** Achieved **83.33%** accuracy on unseen test data using .score(X_test, y_test)
- **Confusion Matrix Insights:**
- **True Positives (Landed correctly):** 12
- **True Negatives (Did not land correctly):** 3
- **False Positives (Predicted landed but didn't):** 3
- **False Negatives:** 0 — indicating **no missed successful landings**, which is crucial for launch reliability

SVM model is **highly sensitive to successful landings** (zero false negatives), which is crucial in mission-critical applications. However, it tends to overpredict landings slightly.

5.BEST EVALUATION RESULT/CLASSIFICATION ACCURACY

TASK 12

Find the method performs best:

In [41]:

```
# Calculate test accuracies
logreg_acc = logreg_cv.score(X_test, Y_test)
svm_acc = svm_cv.score(X_test, Y_test)
tree_acc = tree_cv.score(X_test, Y_test)
knn_acc = knn_cv.score(X_test, Y_test)

# Print accuracies
print(f"Logistic Regression Test Accuracy: {logreg_acc:.4f}")
print(f"SVM Test Accuracy: {svm_acc:.4f}")
print(f"Decision Tree Test Accuracy: {tree_acc:.4f}")
print(f"KNN Test Accuracy: {knn_acc:.4f}")

# Find the best performing model
accuracies = {
    'Logistic Regression': logreg_acc,
    'SVM': svm_acc,
    'Decision Tree': tree_acc,
    'KNN': knn_acc
}

best_model = max(accuracies, key=accuracies.get)
print(f"\nBest performing model: {best_model} with accuracy {accuracies[best_model]:.4f}")
```

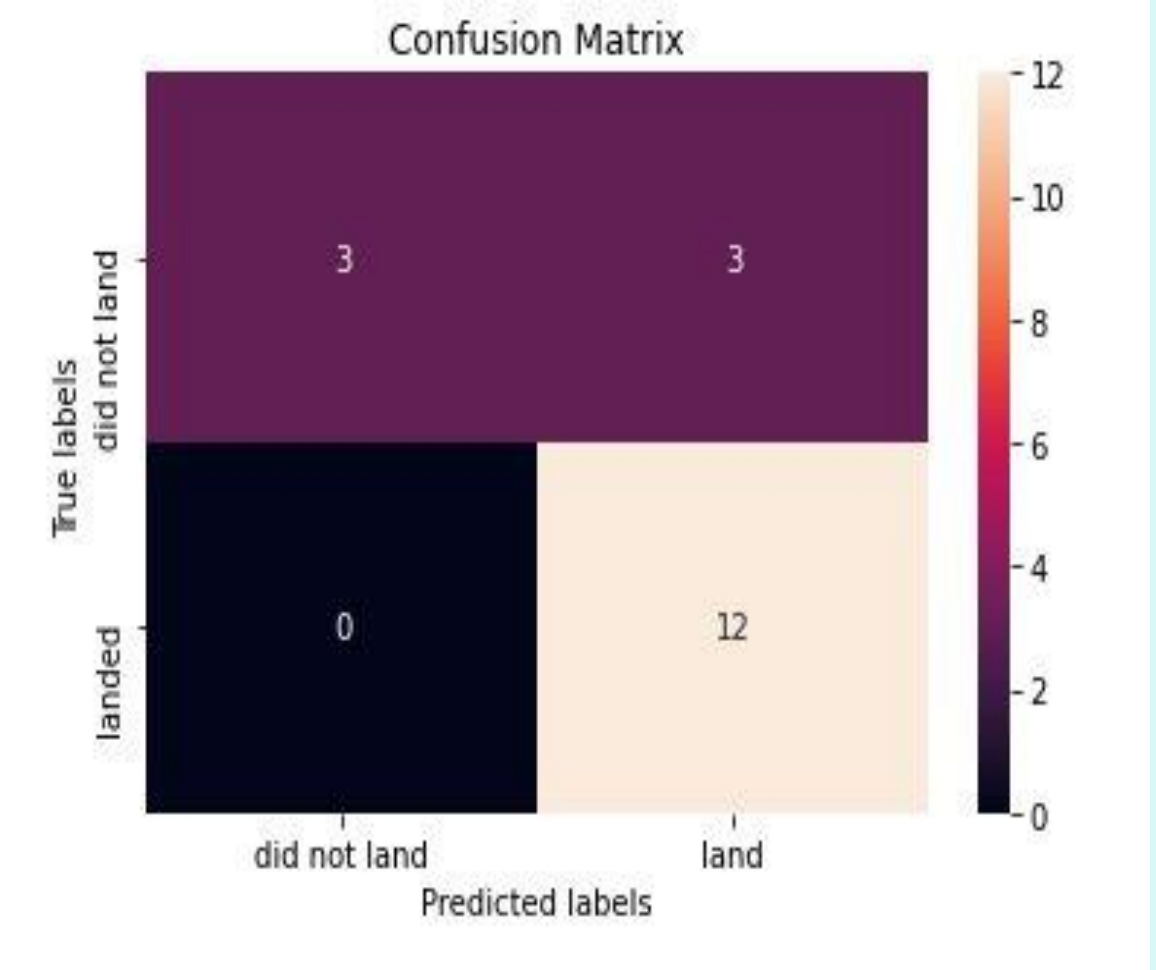
```
Logistic Regression Test Accuracy: 0.8333
SVM Test Accuracy: 0.8333
Decision Tree Test Accuracy: 0.8889
KNN Test Accuracy: 0.8333

Best performing model: Decision Tree with accuracy 0.8889
```

- **The decision tree classifier is the model with the highest classification accuracy**
- LINK TO NOTEBOOK
- https://github.com/Aali715/DATA-SCIENCE-CAPSTONE-PROJECT/blob/250953ca1333d52c0b1b10ac90a8d10882697c7d/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

6. CONFUSION MATRIX OF BEST MODEL DECISION TREE

- 🌳 **Decision Tree Classifier: Confusion Matrix Insights**
- The decision tree model demonstrates a solid ability to distinguish between successful and unsuccessful SpaceX landings, as reflected in the confusion matrix. However, a key issue emerges:
- ⚠**False Positives**
- Definition: Instances where the model incorrectly predicts a successful landing for a launch that actually failed.
- Count: 3 false positives out of 18 total predictions.
- Impact: This type of error can be misleading for mission planning, as it overestimates reliability and may mask operational risks.
- ✅ **Strengths**
- Zero false negatives: The model never misclassified a successful landing as a failure — excellent recall.
- High true positive rate: 12 correct predictions for successful landings.
- 📌 *Conclusion*: While the decision tree classifier is effective at identifying successful landings, its tendency to overpredict success (false positives) highlights a need for improved precision — possibly through hyperparameter tuning or ensemble methods.



CONCLUSION

FINAL INNOVATIVE INSIGHTS & MODEL EVALUATION

Based on our comprehensive analysis of **SpaceX** launch data, we draw the following conclusions:

📈 Launch Trends & Site Performance

Flight Volume vs. Success Rate: Launch sites with higher flight frequencies tend to exhibit greater reliability, suggesting that operational experience and infrastructure maturity contribute to improved outcomes.

Temporal Trend: Launch success rates began rising steadily from **2013**, peaking around **2020**, reflecting technological advancements and process optimization.

Top Performing Site: **KSC LC-39A** recorded the highest number of successful launches, reinforcing its role as SpaceX's most reliable launchpad.

🛰️Orbit Type Reliability

Missions targeting **ES-L1, GEO, HEO, SSO, and VLEO** orbits showed the **highest success rates**, likely due to mature mission profiles and refined targeting protocols.

🤖 Model Selection

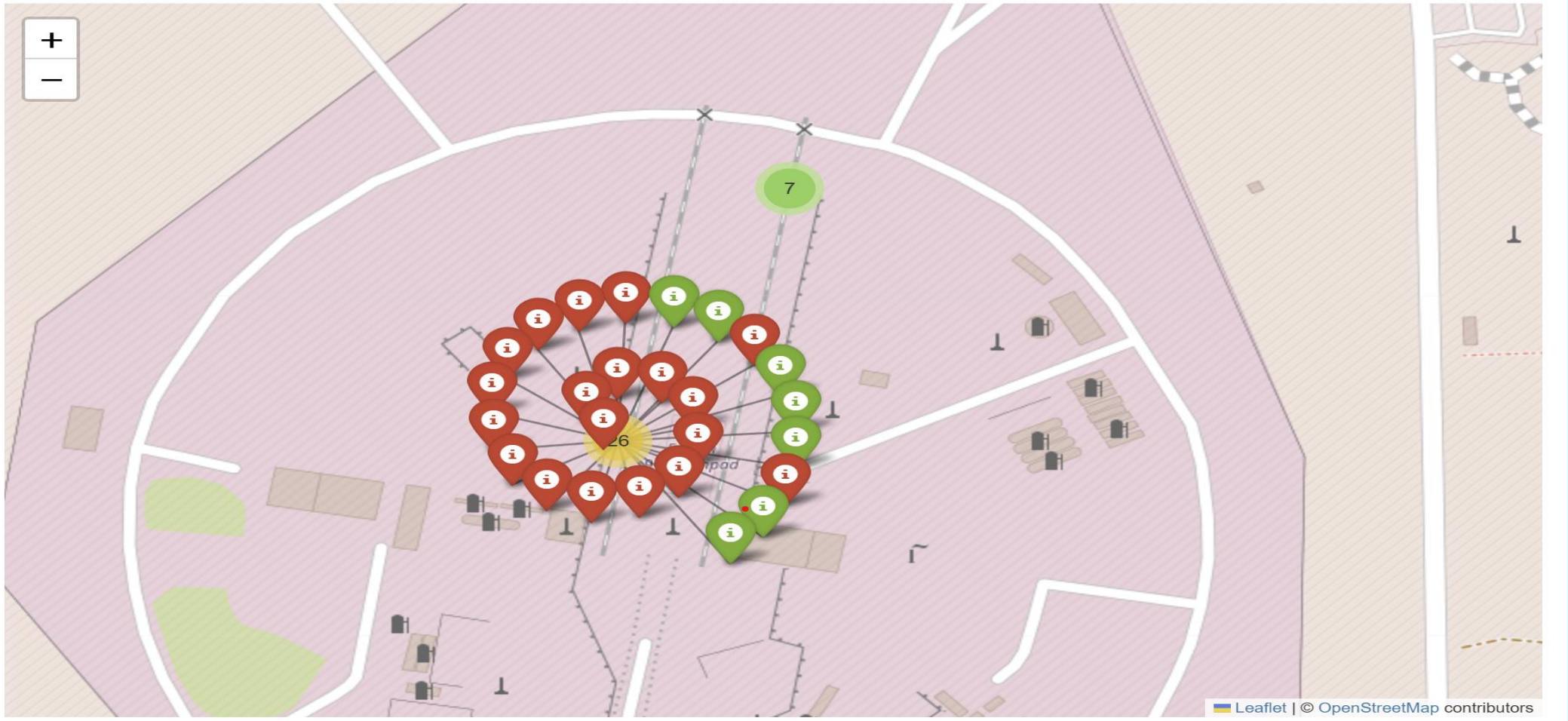
- Among the tested algorithms, the **Decision Tree Classifier** emerged as the most effective for predicting landing outcomes.
- It demonstrated strong class separation and high recall for successful landings.
- Minor issues with false positives suggest potential for further tuning or ensemble enhancement.

📌 ***Conclusion***: **These insights not only inform launch planning and risk assessment but also support cost estimation and competitive bidding strategies against SpaceX.**

APPENDIX

FOLIUM MAP WITH MARKERCLUSTER

13]:



THANK YOU