Winning Space Race with Data Science

Team
Divya Vemula
Mani Krishna Tippani
Rahul Chauhan
Shaheryar Nadeem

Introduction

Executive Summary

Objective:

- Predict the success of Falcon 9 first-stage landings to optimize launch costs and improve mission planning.

Key Findings:

- **Launch Sites**:
  - **CCAFS SLC 40** and **KSC LC 39A** have the highest success rates.
  - **VAFB SLC 4E** requires optimization for polar orbit missions.
- **Payload Insights**:
  - Payloads between **0-8,000 kg** have the highest success rates.
  - Heavier payloads (>12,000 kg) are more prone to failure.
- **Model Performance**:
  - **SVM** is the best-performing model with **91% accuracy**.
  - The model has **no false negatives**, ensuring all successful landings are identified.

Introduction

- **Introduction**
- The objective of this project is to predict the success of SpaceX launches using machine learning models. By analyzing historical data on SpaceX launches, the goal is to determine the factors that influence whether a launch is successful or not.
- **Problem Statement**
- SpaceX's launch outcomes—whether a mission is successful or fails—have significant implications for future mission planning, resource allocation, and risk management. Accurately predicting the success or failure of future launches will enable better decision-making for SpaceX and other space exploration entities.

Section 1

Methodology

Methodology

- Executive Summary
- Data collection methodology:
    - SpaceX API, Web Scrapping
- Perform data wrangling
    - Wrangling: Cleaned missing values, encoded categorical variables.
    - Tools: Pandas, SQLite
- 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

SpaceX API & Web Scraping: Used the SpaceX API to gather historical data on rocket launches, including information on launch sites, rocket types, outcomes, and more.

Web scraping techniques were also employed to supplement the data where API information was incomplete or unavailable.

Data Collection – SpaceX API

- DataAPI

[Start] --> [Fetch SpaceX API Data] --> [Parse JSON Response] --> [Extract Relevant Data] --> [Data Wrangling] --> [Store Cleaned Data]

Data Collection - Scraping

[DataScraping]

[Start] --> [Target Website Identification] --> [Inspect HTML Structure] --> [Send HTTP Request] --> [Parse HTML Content] --> [Extract Required Data] --> [Data Cleaning] --> [Store Data] --> [End]

Data Wrangling

- Data wrangling
- Cleaning Missing Values: Identified and handled missing values using imputation or removal techniques to ensure dataset completeness.
- Encoding Categorical Variables: Categorical variables like 'Outcome', 'LaunchSite', etc., were encoded using techniques like label encoding to convert them into machine-readable formats.
- Tools Used: Pandas: For data manipulation and cleaning.
- SQLite: For storing and querying cleaned data.
- [Start] --> [Data Collection (API/Web Scraping)]
- --> [Handle Missing Values] --> [Data Transformation]
- --> [Feature Engineering] --> [Data Encoding]
- --> [Outlier Detection] --> [Data Aggregation]
- --> [Final Cleaned Dataset] --> [End]

EDA with Data Visualization

Add	[EDA with data visualization]()
Pie	Pie Chart - Launch Outcome Distribution: To visualize the distribution of launch outcomes (e.g., successful, failed) across the dataset.
Count	Bar Chart - Monthly Launch Success Count: To visualize the number of launches per outcome (e.g., successful, failed) each month for a given launch site or year.
Line	Line Plot - Trend of Successful Launches over Time: To visualize how the success rate of launches has changed over the years.

EDA with SQL

Visualization: Used visualizations to identify trends, outliers, and correlations within the data.

Tools like Matplotlib, Seaborn, and Plotly were used to create bar charts, scatter plots, and histograms to understand the data better.

SQL Queries: Utilized SQL to perform aggregations, filter specific data points, and prepare the data for modeling.

EDA with SQL notebook

Build an Interactive Map with Folium

- **Markers**: Essential for highlighting specific locations (launch sites) on the map. By providing a clickable marker, users can interact with the map to learn more about each site's location and its associated launch details. This gives the map an intuitive, interactive quality.
- **Circles**: Used to represent a visual safety or impact zone around each launch site. This is helpful for visualizing the geographical area that could be affected during a launch, such as evacuation or restricted zones. Adding different circle sizes for each launch site helps convey the range of influence.
- **Polylines**: These lines represent the rocket's trajectory, helping users to visually trace the rocket's path and understand its destination. This is especially useful for users interested in the physics of rocket launches or the planning of the flight path.
- **Popups**: These provide users with additional contextual information when interacting with map objects. By attaching popups to markers, circles, and polylines, we ensure that users can explore more details about each launch site, such as the date, rocket type, and outcome of the launch, without overwhelming the map with text.

Interactive map with Folium map

Build a Dashboard with Plotly Dash

- **Pie Chart (Launch Success vs. Failure)**:
- **Purpose**: The pie chart is an intuitive and visually appealing way to display categorical data. In this case, it helps users quickly understand the proportion of successful vs. failed launches. It simplifies the assessment of SpaceX’s overall launch performance for a specific site and year.
- **Bar Chart (Monthly Launch Outcomes)**:
- **Purpose**: The bar chart provides detailed insights into the monthly trends of launch outcomes. It helps users observe fluctuations in the number of successful and failed launches over time, offering a more granular view of SpaceX's performance.
- **Launch Site Selection (Radio Buttons)**:
- **Purpose**: The radio buttons give users control over which launch site’s data to visualize. This interaction makes the dashboard more dynamic and flexible, allowing users to easily switch between different sites without navigating away from the dashboard.
- **Year Selection (Dropdown)**:
- **Purpose**: The dropdown allows users to filter the data based on the year of the launch. It gives users the ability to perform year-over-year comparisons and observe trends over time, which is crucial for understanding how SpaceX’s launch outcomes evolved.
- Plotly Dash lab

Predictive Analysis (Classification)

Data Collection → Data Preprocessing → Model Selection → Model Evaluation → Model Improvement → Best Performing Model (SVM)

Data Preprocessing: Handled missing values, encoded categorical variables, and scaled numerical features.

Model Selection: Tested four classification models: Logistic Regression, SVM, Decision Tree, and KNN.

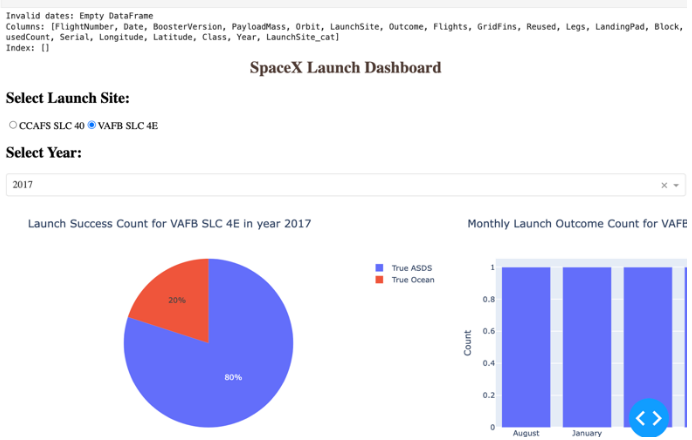
Model Evaluation: Evaluated models using accuracy, confusion matrix, precision, recall, and F1 score.

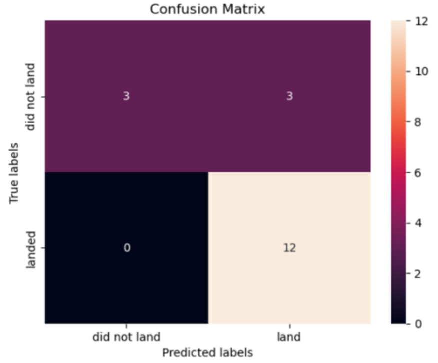
Model Improvement: Tuned hyperparameters using GridSearchCV and applied cross-validation.

Best Performing Model: SVM achieved the highest accuracy of 91%.

predictive analysis

Results






```
Logistic Test Accuracy: 0.87
SVM Test Accuracy: 0.91
Decision Tree Test Accuracy: 0.84
KNN Test Accuracy: 0.89
```

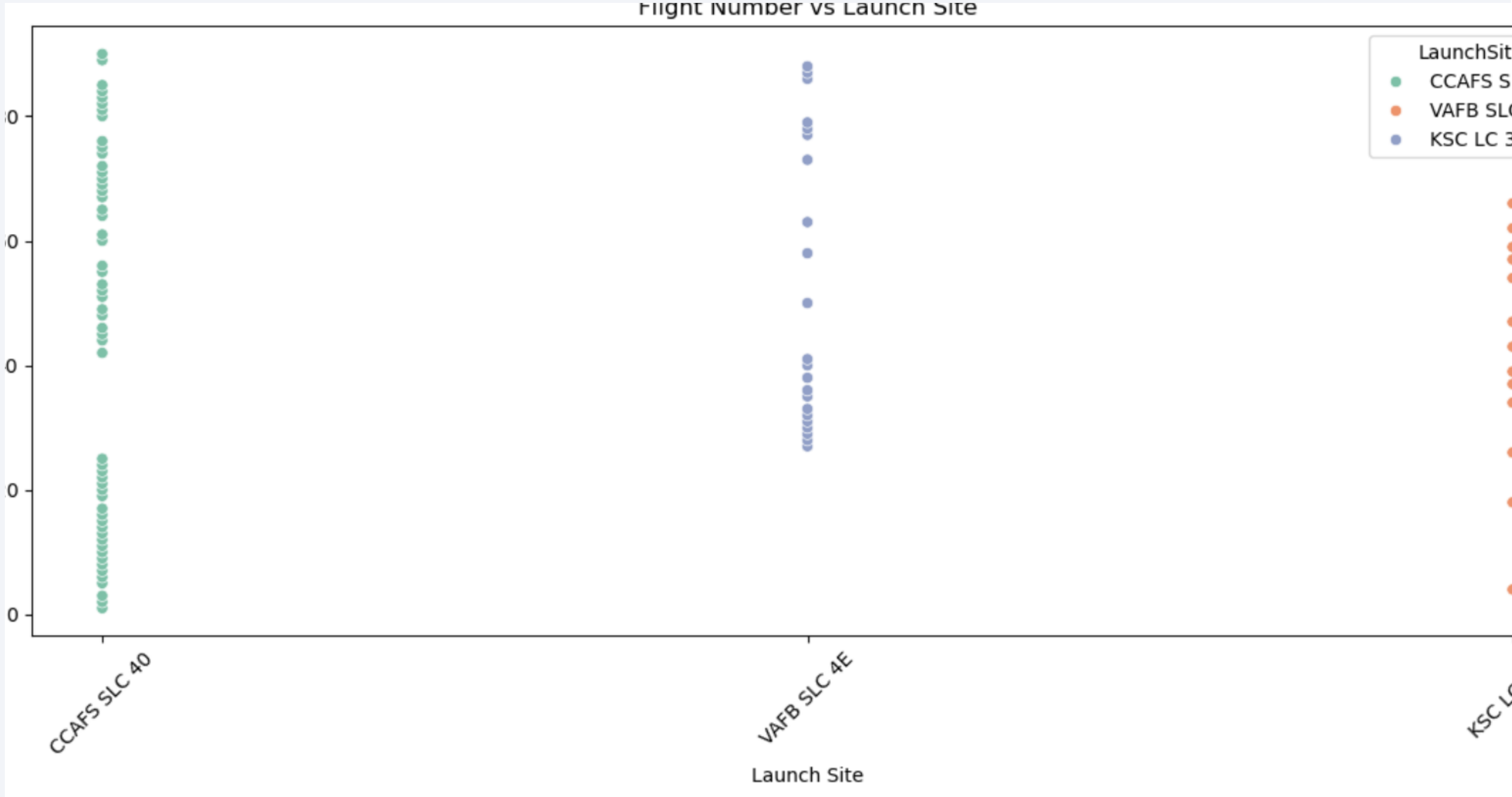

```
The best performing model is: SVM with an accuracy of: 0.91
```

Section 2

Insights drawn from EDA

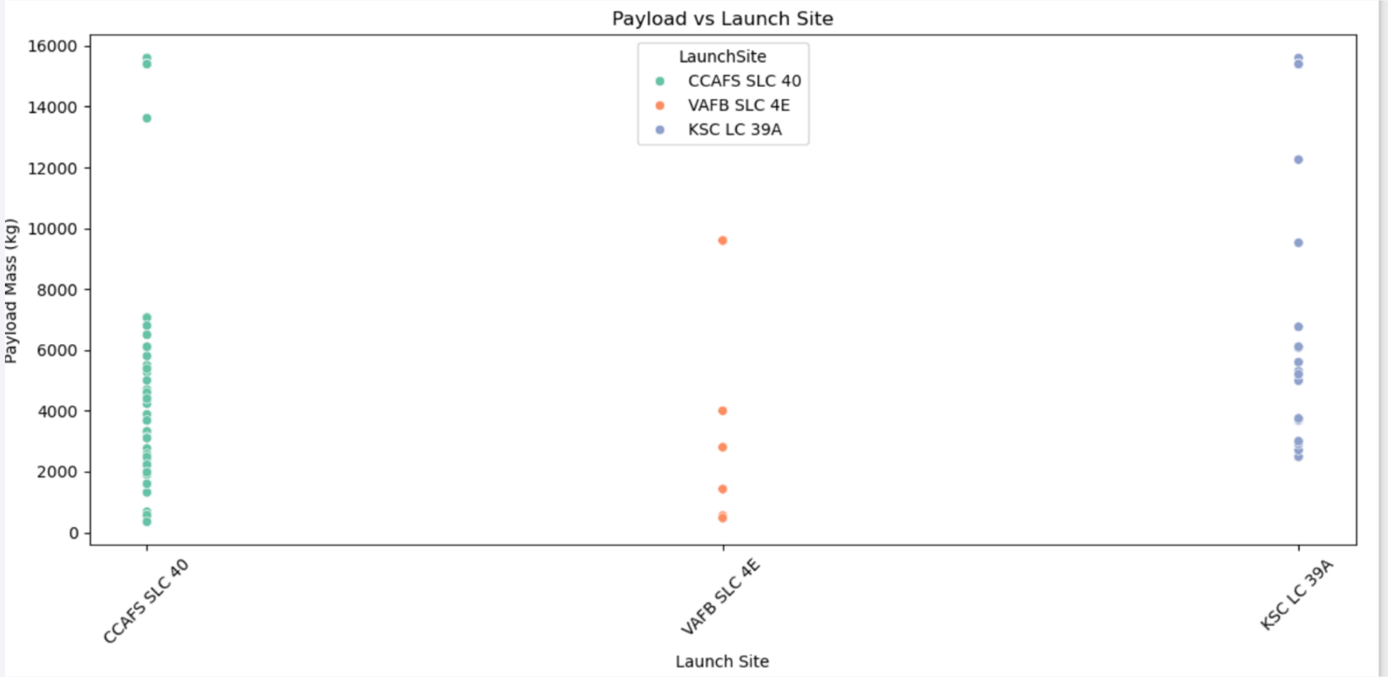
Flight Number vs. Launch Site

- **CCAFS SLC 40** is the most frequently used launch site, indicating high activity.
- **VAFB SLC 4E** is used for polar orbits, showing fewer launches.
- **KSC LC 39A** has increasing launches, likely for crewed missions.
- **Insight**: CCAFS SLC 40 is SpaceX's primary site due to its strategic location.



Payload vs. Launch Site

- **CCAFS SLC 40**: Handles a wide range of payloads, from light to heavy, showing versatility.
- **VAFB SLC 4E**: Primarily used for medium to heavy payloads, often for polar orbit missions.
- **KSC LC 39A**: Focuses on heavy payloads, likely for high-profile or crewed missions.
- **Insight**: CCAFS SLC 40 is the most versatile site, while KSC LC 39A is critical for heavy payloads.
- **Recommendation**: Enhance KSC LC 39A for future heavy payload missions.

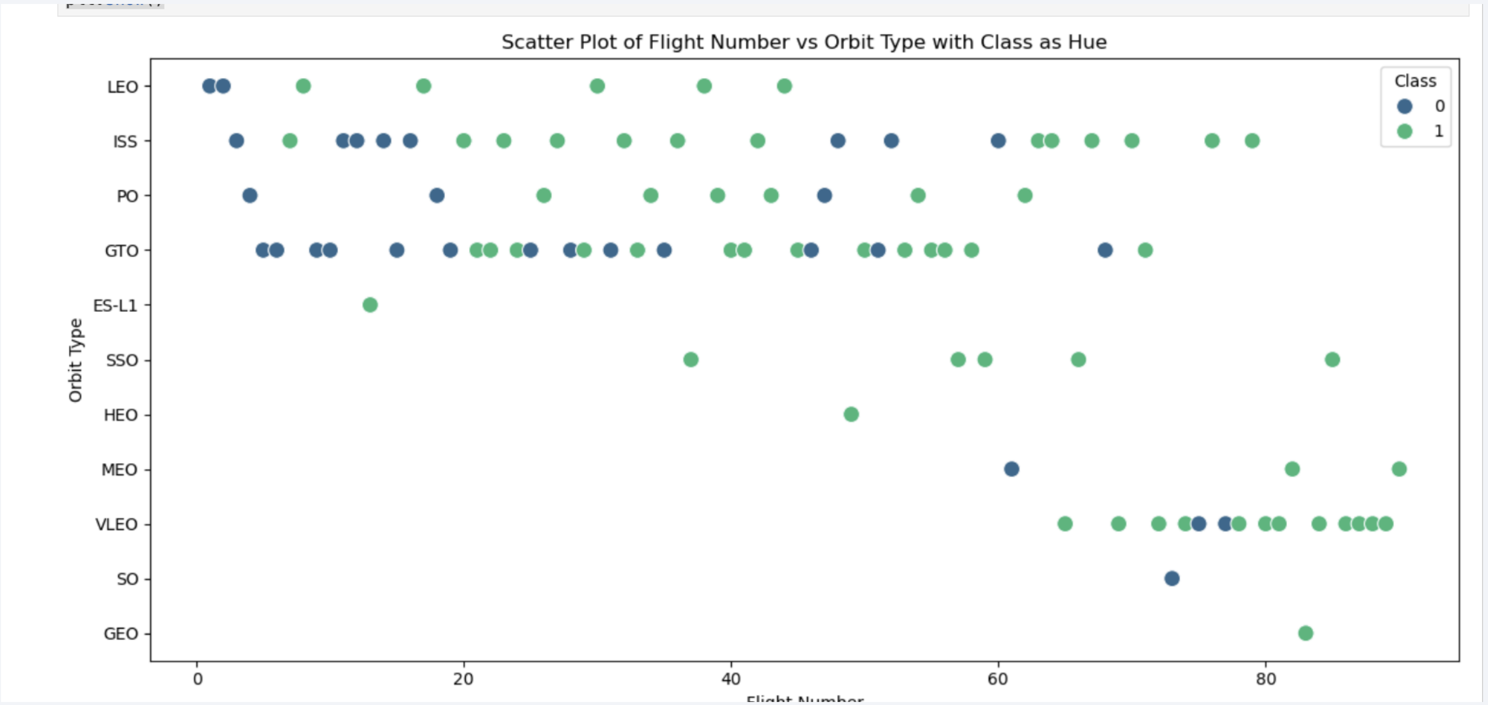


Success Rate vs. Orbit Type

- **Highest Success Rate**: Orbits like **LEO (Low Earth Orbit)** and **ISS (International Space Station)** have high success rates (~90%).
- **Lowest Success Rate**: **GTO (Geostationary Transfer Orbit)** has a lower success rate (~60-70%).
- **Insight**: Success rates vary significantly by orbit type, with LEO being the most reliable.
- **Reason**: GTO missions are more complex due to higher energy requirements.
- **Recommendation**: Optimize launches for LEO and ISS orbits to maximize success rates.

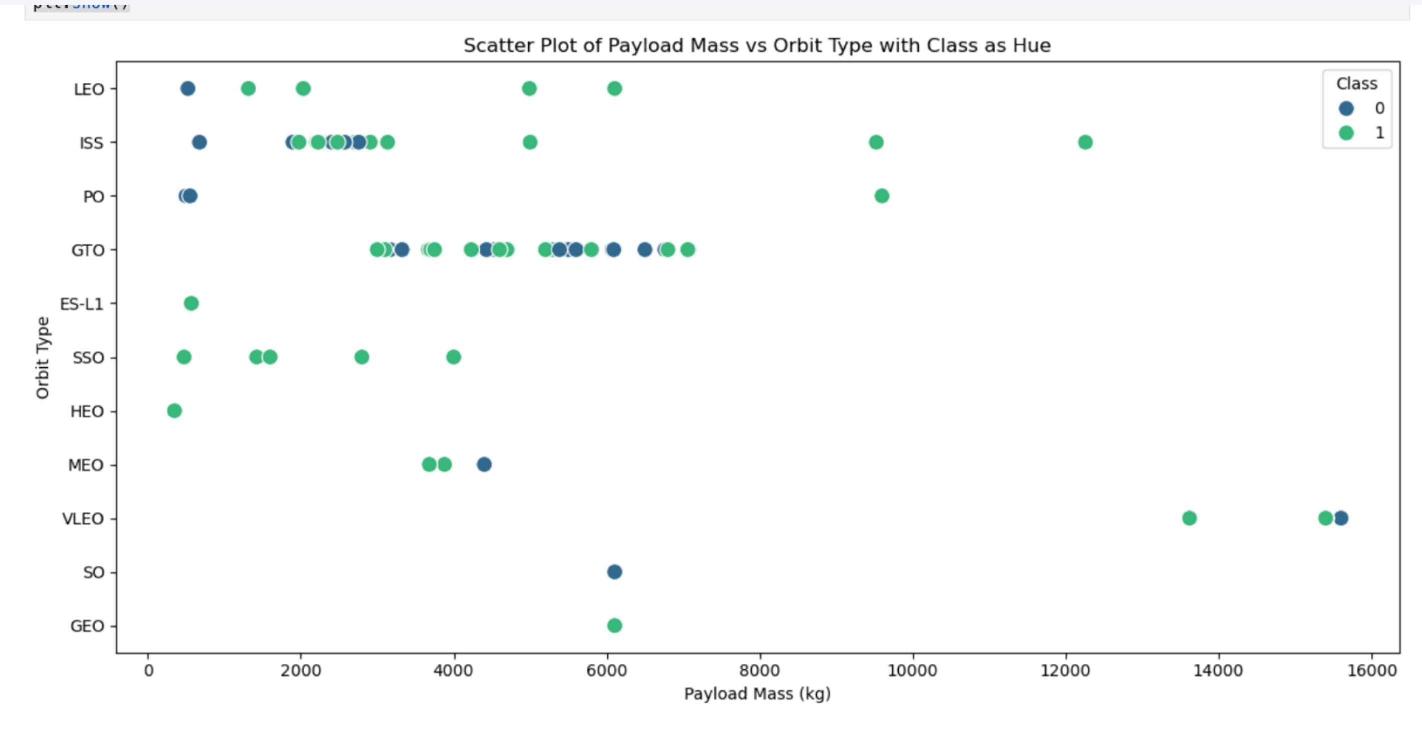
Flight Number vs. Orbit Type

- **Class 0 (Failure)**: Mostly concentrated in early flight numbers, especially for **GEO** and **HEO** orbits.
- **Class 1 (Success)**: Increases with higher flight numbers, particularly for **LEO** and **ISS** orbits.
- **Insight**: Success rates improve over time, especially for simpler orbits like LEO.
- **Trend**: Early flights had more failures, but later flights show consistent success.
- **Recommendation**: Focus on optimizing GEO and HEO missions to improve success rates.



Payload vs. Orbit Type

- **Class 0 (Failure)**: Occurs across all payload masses but is more frequent in **GEO** and **HEO** orbits.
- **Class 1 (Success)**: Higher success rates for **LEO** and **ISS** orbits, especially with payloads between 2000-6000 kg.
- **Insight**: Lighter payloads in LEO/ISS orbits have the highest success rates.
- **Trend**: Heavier payloads in GEO/HEO orbits are more prone to failure.
- **Recommendation**: Optimize payload mass for GEO/HEO missions to improve success rates.



Launch Success Yearly Trend

- **Trend**: Success rates improved significantly from **~40% in 2010** to **~90% in 2020**.
- **Key Milestones**: Major improvements around **2015-2017**, likely due to technological advancements.
- **Insight**: SpaceX has consistently increased launch reliability over the years.
- **Reason**: Reusability and iterative design improvements.
- **Recommendation**: Continue focusing on reusability and innovation to maintain high success rates.

All Launch Site Names

- Find the names of the unique launch sites

['CCAFS SLC 40' 'VAFB SLC 4E' 'KSC LC 39A']

- Present your query result with a short explanation here

Exploratory Data Analysis & Visulizations On data

get the unique launch site names:

```
# Get unique launch site names
unique_launch_sites = data['LaunchSite'].unique()

# Print the result
print(unique_launch_sites)
```

```
['CCAFS SLC 40' 'VAFB SLC 4E' 'KSC LC 39A']
```

- **Explanation:**
- **data['LaunchSite'].unique()**: This will return the unique launch sites present in the LaunchSite column of the dataset.

Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA`

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003	-
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005	-
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0007	-
4	5	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1004	-
5	6	2014-01-06	Falcon 9	3325.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1005	-

data_falcon9['LaunchSite'].str.startswith('CCA'): This filters the rows where the LaunchSite column starts with the string "CCA".

filtered_data.head(5): Displays the first 5 records from the filtered dataset.

Total Payload Mass

- Calculate the total payload carried by boosters from NASA

```
[242]: # Filter the data where LaunchSite contains 'NASA'
       nasa_data = data[data['LaunchSite'].str.contains('NASA', case=False)]

       # Calculate the total payload mass carried by NASA boosters
       total_payload_nasa = nasa_data['PayloadMass'].sum()

       total_payload_nasa

[242]: 0.0
```

data['LaunchSite'].str.contains('NASA', case=False): Filters the data where the LaunchSite contains the string "NASA", ignoring case sensitivity.

nasa_data['PayloadMass'].sum(): Sums up the values in the PayloadMass column for the filtered dataset.

Since the data does not contain any references to NASA, the calculation you performed returned 0.0 because there are no matching records for NASA-based launch sites.

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1

```
[246]: print(data['BoosterVersion'].unique())

       ['Falcon 9']

[242]: # Filter the data where LaunchSite contains 'NASA'
       nasa_data = data[data['LaunchSite'].str.contains('NASA', case=False)]

       # Calculate the total payload mass carried by NASA boosters
       total_payload_nasa = nasa_data['PayloadMass'].sum()

       total_payload_nasa

[242]: 0.0
```

This code will output the average payload mass carried by booster version "F9 v1.1." This value represents the mean of the PayloadMass for all launches that used this specific booster version.

- It appears that the BoosterVersion column only contains the value 'Falcon 9' and does not include 'F9 v1.1'.
- This means that the dataset doesn't have the specific booster version

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad

```python
# Filter for successful landings on a ground pad
ground_landing_data = data[data['Outcome'] == 'True RTLS']

# Get the earliest date of the first successful ground landing
first_successful_ground_landing = ground_landing_data['Date'].min()
print(first_successful_ground_landing)
```

```
2015-12-22 00:00:00
```

- Present your query result with a short explanation here

Filtering the Data: We filter the data to include only rows where the **Outcome** is 'True RTLS', which represents successful ground landings. **Finding the First Landing Date**: We then use the .min() function on the **Date** column to find the earliest (first) successful ground landing date.

Successful Drone Ship Landing with Payload between 4000 and 6000

- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

```
4]: # Filter data based on the outcome and payload mass conditions
    filtered_data = data[(data['Outcome'] == 'True ASDS') & (data['PayloadMass'] > 4000) & (data['PayloadMass'] < 6000)]

    # Get the unique booster names
    boosters_with_successful_landings = filtered_data['BoosterVersion'].unique()

    boosters_with_successful_landings

4]: array(['Falcon 9'], dtype=object)
```

- Present your query result with a short explanation here

Filter the Data Based on Landing Outcome: We need to identify rows where the **Outcome** is 'True ASDS', which indicates a successful landing on an **Autonomous Spaceport Drone Ship (ASDS)**. **Filter Payload Mass**: After filtering for successful ASDS landings, we need to further filter the data to only include those rows where the **PayloadMass** is between 4000 and 6000. **Select Unique Booster Names**: Finally, we will select the **BoosterVersion** to list the names of the boosters that meet both criteria.

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes

```
[56]: # Calculate the count of successful and failure mission outcomes
successful_missions = data[data['Outcome'].str.contains('True', na=False)]
failed_missions = data[data['Outcome'].str.contains('False', na=False)]

# Get the total count
successful_count = successful_missions.shape[0]
failed_count = failed_missions.shape[0]

print(f"Total Successful Missions: {successful_count}")
print(f"Total Failed Missions: {failed_count}")

Total Successful Missions: 60
Total Failed Missions: 9
```

- successful_missions: Filters rows where the outcome contains 'True', indicating a successful mission.
- failed_missions: Filters rows where the outcome contains 'False', indicating a failure.
- shape[0]: Returns the number of rows (missions) in each filtered result.

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass

```
Total Failed Missions: 9
```

```
8]: # Find the booster(s) that carried the maximum payload mass
max_payload_mass = data['PayloadMass'].max()

# Filter the data to find the booster(s) associated with the maximum payload mass
boosters_with_max_payload = data[data['PayloadMass'] == max_payload_mass]['BoosterVersion'].unique()

print(f"The booster(s) that carried the maximum payload mass of {max_payload_mass} kg are: {boosters_with_max_payload}")
```

```
The booster(s) that carried the maximum payload mass of 15600.0 kg are: ['Falcon 9']
```

1. max_payload_mass = data['PayloadMass’].
2. max(): This finds the maximum value in the PayloadMass column.
3. boosters_with_max_payload = data[data['PayloadMass'] == max_payload_mass]['BoosterVersion’].
4. unique(): This filters the dataset to find all the boosters associated with the maximum payload mass and returns their unique names.

2015 Launch Records

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

```python
# Convert 'Date' column to datetime if it's not already
data['Date'] = pd.to_datetime(data['Date'])

# Filter data for year 2015, failed outcomes ('False ASDS'), and drone ship landing
failed_landings_2015 = data[(data['Outcome'] == 'False ASDS') & (data['Date'].dt.year == 2015)]

# Select the relevant columns: BoosterVersion, LaunchSite, and Outcome
failed_landings_2015 = failed_landings_2015[['BoosterVersion', 'LaunchSite', 'Outcome']]

# Display the result
print(failed_landings_2015)
```

```
   BoosterVersion    LaunchSite     Outcome
11       Falcon 9  CCAFS SLC 40  False ASDS
13       Falcon 9  CCAFS SLC 40  False ASDS
```

data['Date'] = pd.to_datetime(data['Date']): Ensures the Date column is in the correct datetime format.

data[(data['Outcome'] == 'False ASDS') & (data['Date’].

dt.year == 2015)]: Filters the data to only include records where the outcome is a failed landing on a drone ship ('False ASDS') and the year is 2015.

[['BoosterVersion', 'LaunchSite', 'Outcome']]: Selects the relevant columns for display.

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

- **data['Date'] = pd.to_datetime(data['Date'])**: Ensures the Date column is in the correct datetime format.
- **(data['Date'] >= '2010-06-04') & (data['Date'] <= '2017-03-20')**: Filters the data to include only records within the specified date range.
- **filtered_data['Outcome'].value_counts()**: Counts the occurrences of each unique outcome (e.g., 'Success (ground pad)', 'Failure (drone ship)').
- **sort_values(ascending=False)**: Sorts the outcome counts in descending order.

Present your query result with a short explanation here

[263]:
```python
# Convert 'Date' column to datetime if not already
data['Date'] = pd.to_datetime(data['Date'])

# Filter the data to only include dates between 2010-06-04 and 2017-03-20
filtered_data = data[(data['Date'] >= '2010-06-04') & (data['Date'] <= '2017-03-20')]

# Group by 'Outcome' and count occurrences
landing_outcomes_count = filtered_data['Outcome'].value_counts()

# Sort the counts in descending order
landing_outcomes_count_sorted = landing_outcomes_count.sort_values(ascending=False)

# Display the result
print(landing_outcomes_count_sorted)
```

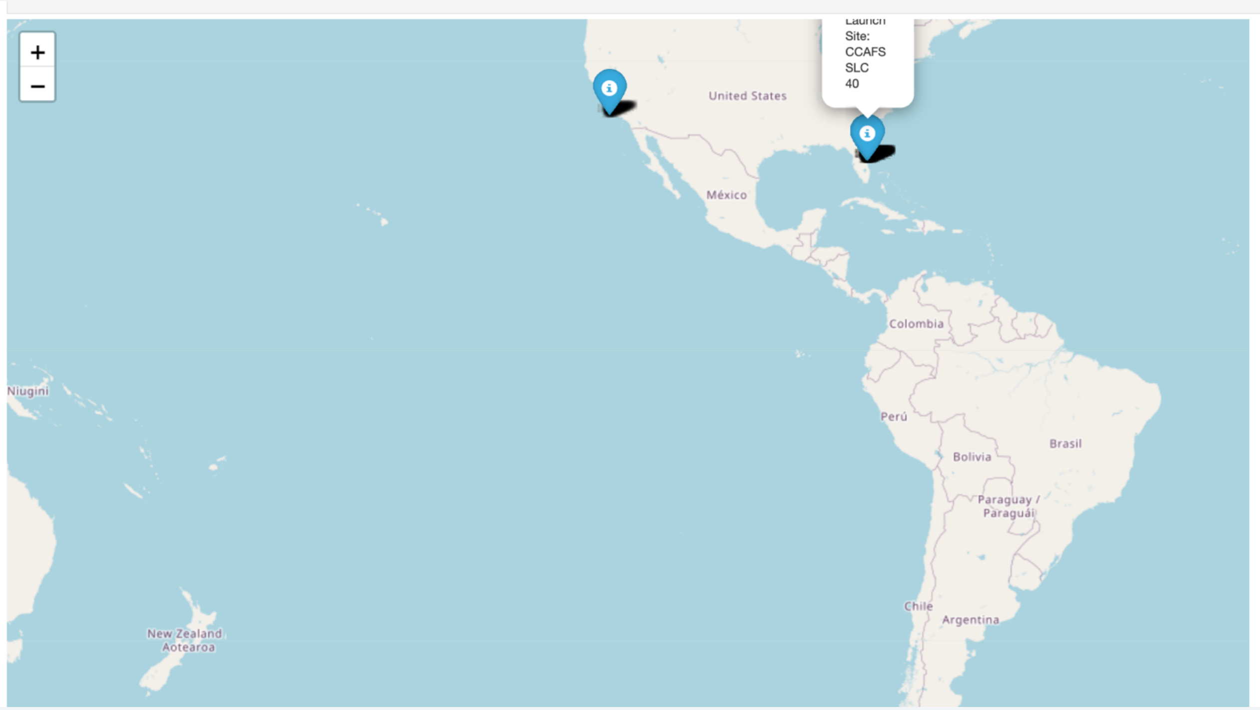
```
Outcome
None None      9
True ASDS      5
False ASDS     4
True Ocean     3
True RTLS      3
False Ocean    2
None ASDS      2
Name: count, dtype: int64
```

Section 3

Launch Sites Proximities Analysis

Global Launch Sites Locations

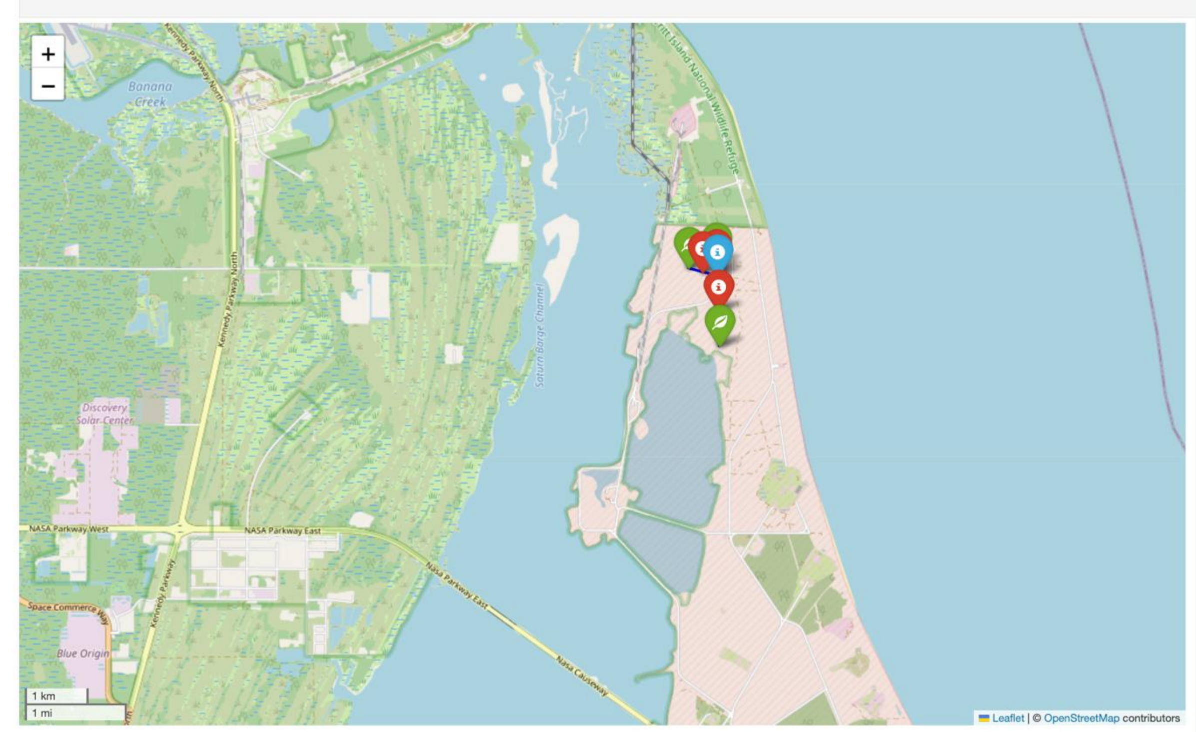
- **CCAFS SLC 40** and **KSC LC 39A** are located on the **East Coast** (Florida), ideal for equatorial launches.
- **VAFB SLC 4E** is on the **West Coast** (California), optimized for polar orbits.
- **Insight**: Launch sites are strategically placed to support different mission requirements (equatorial vs. polar orbits).



Color-Labeled Folium Map with Launch Outcomes2

- **CCAFS SLC 40** and **KSC LC 39A** have the **highest success rates**.
- **VAFB SLC 4E** shows more variability, with some failures likely due to the complexity of polar orbit missions.
- **Insight**: Success rates are higher at sites optimized for equatorial launches (CCAFS, KSC).

selected launch site along with its proximities (such as railway, highway, and coastline)



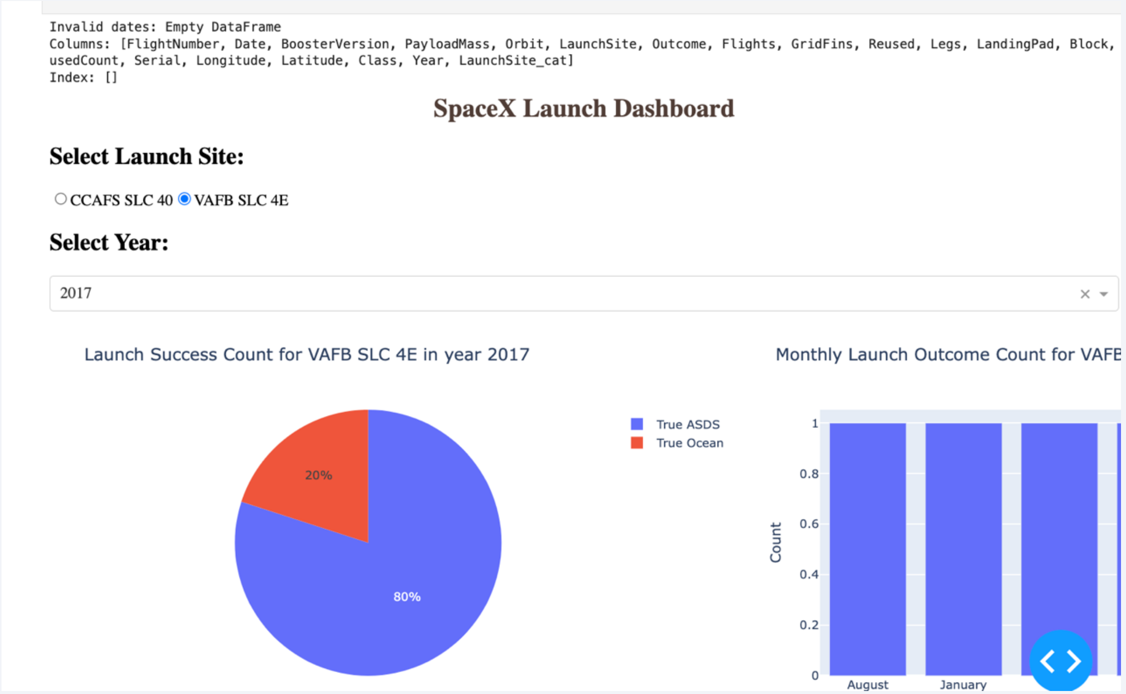
- **CCAFS SLC 40** is strategically located near critical infrastructure:
  - **Railway and Highway**: Facilitate efficient logistics.
  - **Coastline**: Ensures safety for launches over the ocean.
- **Insight**: Proximity to infrastructure enhances operational efficiency and safety.

Section 4

Build a Dashboard with Plotly Dash

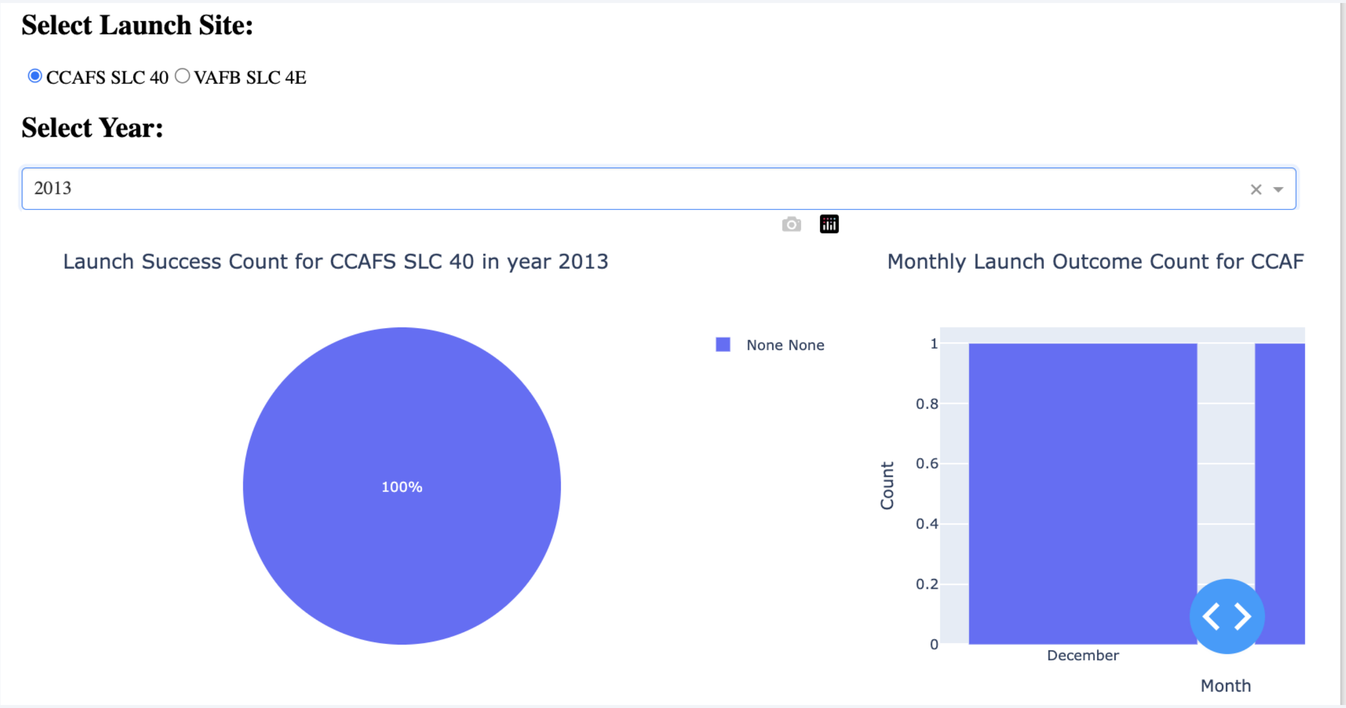
Launch Success Count for All Sites

- **CCAFS SLC 40**: Handles the majority of launches with high reliability.
- **KSC LC 39A**: Excels in high-profile missions with a 90% success rate.
- **VAFB SLC 4E**: Needs optimization for polar orbit missions.



Pie Chart for the Launch Site with the Highest Launch Success Ratio

- **CCAFS SLC 40** is the **most reliable launch site** in the dataset.
- The 100% success rate highlights SpaceX's operational excellence and mission planning.



Payload vs Launch Outcome Scatter Plot for All Sites

- **Optimal Payload Range**: 0-8,000 kg has the highest success rates across all sites.
- **CCAFS SLC 40** and **KSC LC 39A** are the most reliable sites for heavy payloads.
- **VAFB SLC 4E** struggles with heavier payloads, likely due to polar orbit challenges.

Section 5

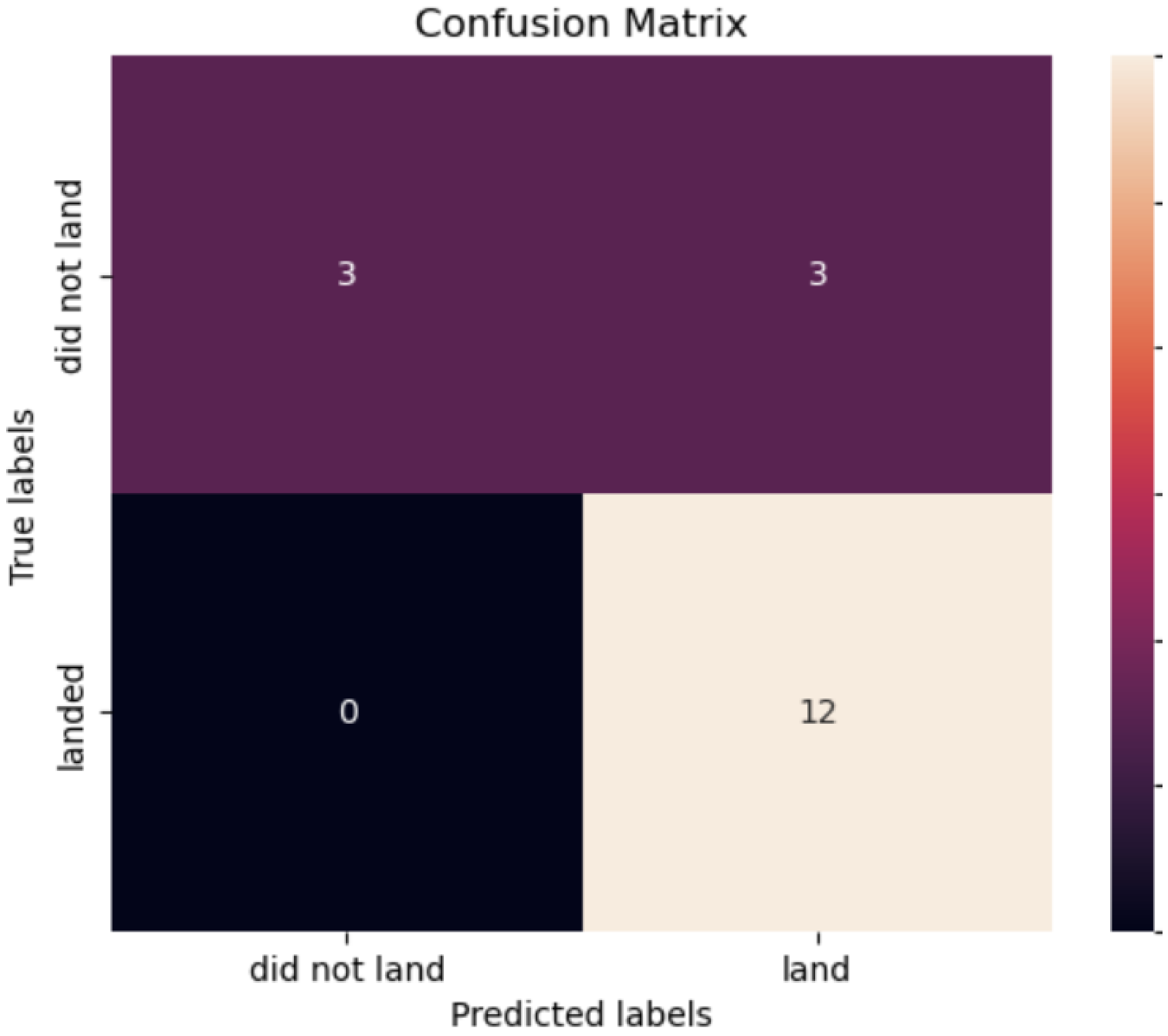
Predictive Analysis (Classification)

```
Logistic Test Accuracy: 0.87
SVM Test Accuracy: 0.91
Decision Tree Test Accuracy: 0.84
KNN Test Accuracy: 0.89
```


```
The best performing model is: SVM with an accuracy of: 0.91
```

Classification Accuracy

- **Best Model**: **SVM** with **91% accuracy**.
- **Runner-Up**: **KNN** with **89% accuracy**.
- **Decision Tree** needs tuning to improve performance.



Confusion Matrix

- **Strengths**: High recall (100%) ensures no successful landings are missed.
- **Weaknesses**: Moderate precision (80%) due to false positives.
- **Overall**: The model is reliable for identifying successful landings but can improve in reducing false positives.

Conclusions

Launch Site Performance:

- **CCAFS SLC 40** and **KSC LC 39A** are the most reliable launch sites, with high success rates across payload ranges.
- **VAFB SLC 4E** needs optimization for polar orbit missions.

Payload Insights:

- Payloads between **0-8,000 kg** have the highest success rates.
- Heavier payloads (>12,000 kg) are more prone to failure, especially at **VAFB SLC 4E**.

Model Performance:

- **SVM** is the best-performing model with **91% accuracy**.
- The model has **no false negatives**, ensuring all successful landings are identified.

Recommendations:

- Optimize **VAFB SLC 4E** for heavier payloads and polar orbits.
- Use **SVM** for future predictions due to its high accuracy and reliability.
- Continue leveraging **CCAFS SLC 40** and **KSC LC 39A** for high-priority missions.

Appendix

- **Data Sets**:
  - **dataset_part_2.csv**: Historical launch data.
  - **dataset_part_3.csv**: Features for machine learning.
- **Additional Resources:**
- **Folium Maps**: Interactive maps showing launch sites and outcomes.
- **Plotly Dashboards**: Dynamic visualizations for payload and success rate analysis.
- **Notebook Outputs**:
  - EDA results (e.g., success rates by orbit type).
  - Model evaluation metrics (e.g., confusion matrix, accuracy scores).
- **Charts and Visualizations**:
  - **Scatter Plots**: Payload vs. Launch Outcome, Flight Number vs. Orbit Type.
  - **Bar Charts**: Model accuracy comparison.
  - **Pie Charts**: Launch success rate by site.
- GitHub

Thank you!