



IBM Developer  
SKILLS NETWORK

# Winning Space Race with Data Science

Kevin Price  
June 12, 2024



# Outline

---

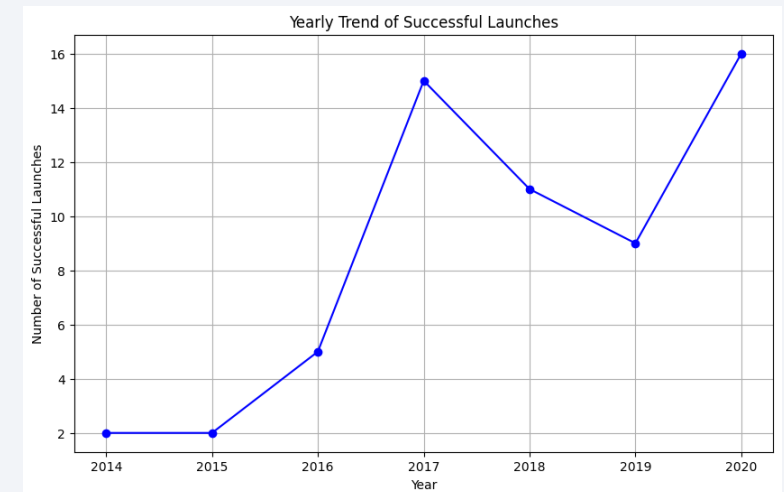
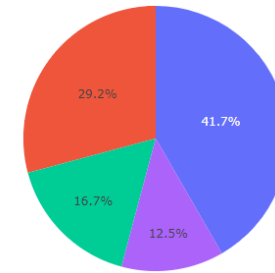
- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion
- Appendix

# Executive Summary

- Using many data collection, wrangling, and analysis techniques we determined the best way to predict the successful first stage landing of a Falcon 9 rocket and provide a dashboard for additional analysis.
- To do this we used the SpaceX API, webscraping, regression, and visualization methodologies to review historical launch data and make a decision.
- The results show that:
  - There is a positive relationship for LEO orbit and no relationship for GTO orbit
  - There is a positive relationship between payload weight and Polar, LEO, and ISS
  - Using this information, successful launches have increased from 2014 to 2020

- Summary of methodologies
- Summary of all results

SpaceX Launch Records Dashboard1



# Introduction

---

- Project background and context
- Problems you want to find answers

- The SpaceX program aims to provide an economical edge in getting rockets into space. To do this, we have to analyze each aspect of the launch process to determine what works and what doesn't.
- To do this, we look at some of questions that need answers:
  - How many launches has SpaceX conducted each year?
  - What are the most common launch sites used by SpaceX?
  - What is the distribution of payload masses across all launches?
  - Which orbits are most frequently targeted by SpaceX launches?
  - What is the success rate of different booster versions?
  - How many times has each booster been reused?
  - Are there any noticeable trends or patterns in launch outcomes over time?
  - How has the success rate of launches evolved over the years?
  - How do launch outcomes vary by launch site locations (latitude and longitude)?
  - Can we predict the success of a launch based on features like payload mass, booster version, and launch site?



Section 1

# Methodology

# Methodology

---

## Executive Summary

- Data collection methodology:
  - Data sources: APIs, Web Scraping, Databases, and Public Datasets
  - Tools used: Python (requests, BeautifulSoup), SQL
- Perform data wrangling
  - Data Cleaning: Handling missing values, removing duplicates
  - Data Transformation: Normalization, Scaling
- Perform exploratory data analysis (EDA) using visualization and SQL
  - Techniques: Visualization (Matplotlib, Seaborn), SQL Queries
  - Insights Gained: Identified patterns and trends, Detected anomalies

# Methodology Continued

---

- Perform interactive visual analytics using Folium and Plotly Dash
  - Tools: Folium (interactive maps), Plotly Dash (dashboards)
  - Applications: Geospatial Analysis, Dynamic data exploration
- Perform predictive analysis using classification models
  - Models used: Regression, SVM, Decision Trees, KNN
  - Evaluation Metrics: Accuracy, Precision, Recall
- How to build, tune, evaluate classification models
  - Steps: Model selection, Hyperparameter Tuning (GridSearchCV), Cross-validation
  - Tools: Scikit-learn, GridSearchCV

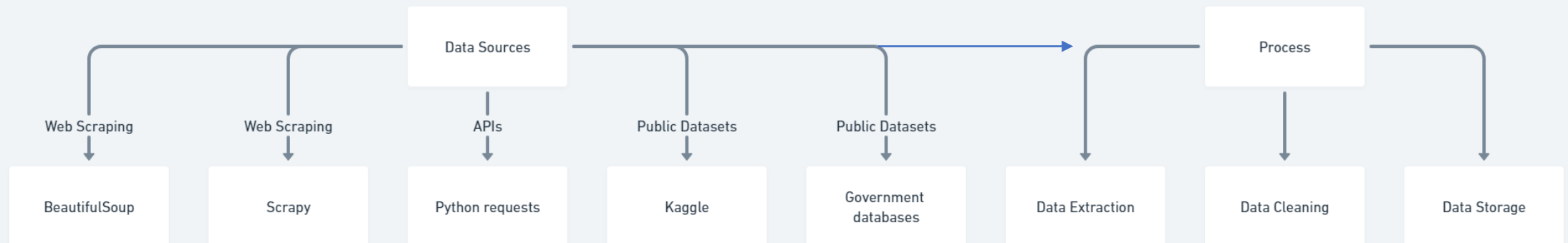
# Data Collection

---

Web Scraping was used through automated data extraction from websites with tools BeautifulSoup and Scrapy. For example: Collecting launch details from SpaceX's official website.

APIs were used through the Python requests library. For example: Gathering real-time launch data from NASA's API.

Utilized openly available datasets like Kaggle and government databases. For example: Historical launch data from Kaggle.



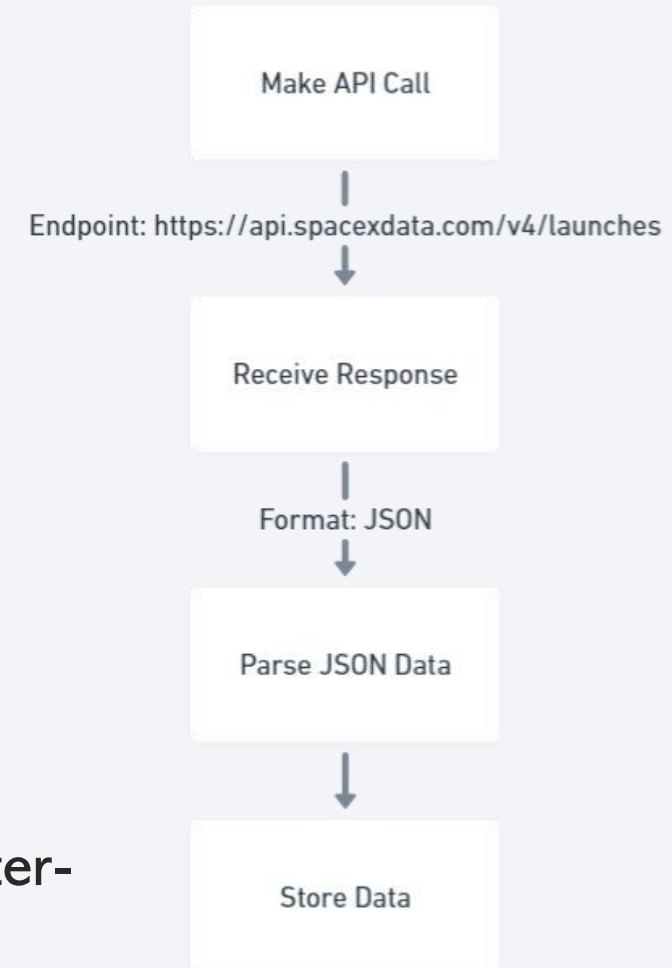


# Data Collection – SpaceX API

- REST API
  - Using SpaceX's public REST API to collect launch data to access endpoints for different data types: launches, rockets, payloads.
- Tools –
  - Python requests library to make API calls.
  - JSON handling o parse and store data.
- Example Endpoints:
  - Launch Data: <https://api.spacexdata.com/v4/launches>
  - Rocket Data: <https://api.spacexdata.com/v4/rockets>
  - Payload Data: <https://api.spacexdata.com/v4/payloads>

Repository can be located here:

- <https://github.com/kpcrash/testrepo/blob/main/jupyter-labs-spacex-data-collection-api.ipynb>



# Data Collection - Scraping

- Identify Target Website
  - Determine the website to scrape.
  - Example: SpaceX launch data from the SpaceX official website.
- Inspect Web Page
  - Use browser developer tools to inspect the HTML structure.
  - Identify the HTML tags and classes containing the desired data.
- Extract Data
  - Use web scraping libraries to extract data.
  - Tools: BeautifulSoup, Scrapy. - Parse HTML to locate and extract relevant information.
- Store Data –
  - Save the extracted data in a structured format.
  - Formats: CSV, database.

Repository can be located here:

- <https://github.com/kpcrash/testrepo/blob/main/jupyter-labs-webscraping.ipynb>



# Data Wrangling

- Data Cleaning

- Handle missing values (e.g., imputation, removal).
- Remove duplicates to ensure data integrity.
- Correct data types (e.g., converting strings to dates).

- Data Transformation

- Normalize data to ensure consistency.
- Scale features to standardize ranges.
- Encode categorical variables (e.g., one-hot encoding).

- Data Integration

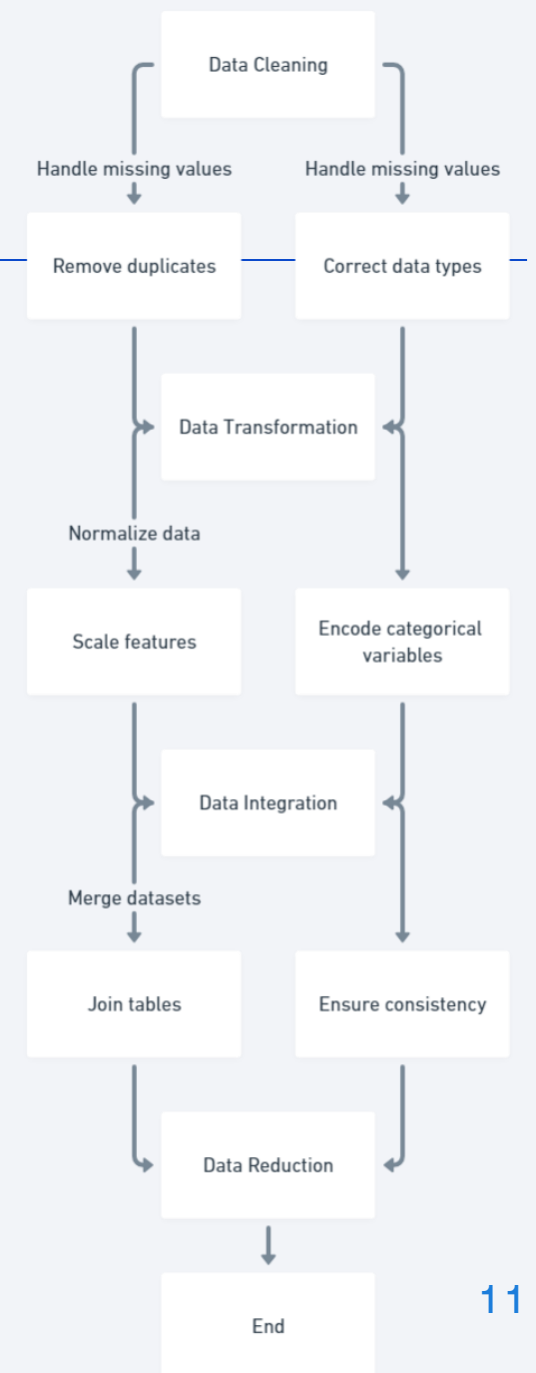
- Merge datasets from multiple sources.
- Join tables based on common keys.
- Ensure consistency and accuracy across datasets.

- Data Reduction

- Select relevant features for analysis.
- Remove irrelevant or redundant data.
- Aggregate data to reduce dimensionality.

- Example Repositories showing data wrangling:

- <https://github.com/kpcrash/testrepo/blob/main/jupyter-labs-spacex-data-collection-api.ipynb>
- <https://github.com/kpcrash/testrepo/blob/main/jupyter-labs-webscraping.ipynb>
- <https://github.com/kpcrash/testrepo/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb>



# EDA with Data Visualization

---

- Exploratory Data Analysis (EDA) helps in understanding the data by summarizing its main characteristics often using visual methods. Data visualization aids in identifying patterns, anomalies, and relationships within the data.
- We used the following types of charts in support of this effort:
  - Histogram to help in understanding the spread and central tendency of the data
  - Box Plot to highlight the median, quartiles, and potential outliers
  - Scatter Plot to identify trends, correlations, and patterns between variables
  - Bar Chart to compare and visualize categorical data
  - Pie Charts to show a visual representation of the part-to-whole relationships of data
  - Line Charts for showing changes and trends over time
- EDA Visualization Repository:
  - <https://github.com/kpcrash/testrepo/blob/main/edadataviz.ipynb>

# EDA with SQL

---

- Create Table: Created a table to store launch data.
- Insert Data: Inserted records into the table for analysis.
- Select Queries: Retrieved all records, specific columns, and filtered data based on conditions.
- Aggregate Functions: Used functions like COUNT, SUM, and AVG to summarize data.
- Joins: Combined data from multiple tables to enrich the dataset.
- Group By: Grouped data to compute statistics for different categories.
- Order By: Sorted data based on specific columns for better readability.
- EDA with SQL Notebook is located here:
  - [https://github.com/kpcrash/testrepo/blob/main/jupyter-labs-eda-sql-coursera\\_sqlite.ipynb](https://github.com/kpcrash/testrepo/blob/main/jupyter-labs-eda-sql-coursera_sqlite.ipynb)



# Build an Interactive Map with Folium

---

Object Type	Purpose	Why Used
Markers	Indicate specific locations on the map	Identify locations of launch sites
Circles	Highlight areas around points	Show spatial relation of launch sites
Lines	Connect multiple points on a map	Visualize connections between points

- Folium map repository is located here:
  - [https://github.com/kpcrash/testrepo/blob/main/lab\\_jupyter\\_launch\\_site\\_location.ipynb](https://github.com/kpcrash/testrepo/blob/main/lab_jupyter_launch_site_location.ipynb)

# Build a Dashboard with Plotly Dash

---

- A Plotly Dashboard was created to allow for interactivity, clarity, and flexibility allowing users to drill down on details.
- Dashboard contains a dropdown selector that influences a pie chart and range slider that filters data based on payload mass range shown in a scatter plot
- Both dropdown and range slider inputs dynamically update the scatter plot to reflect selected filters.
- Plotly Dash repository is located here:
  - [https://github.com/kpcrash/testrepo/blob/main/spacex\\_dash\\_app.py](https://github.com/kpcrash/testrepo/blob/main/spacex_dash_app.py)

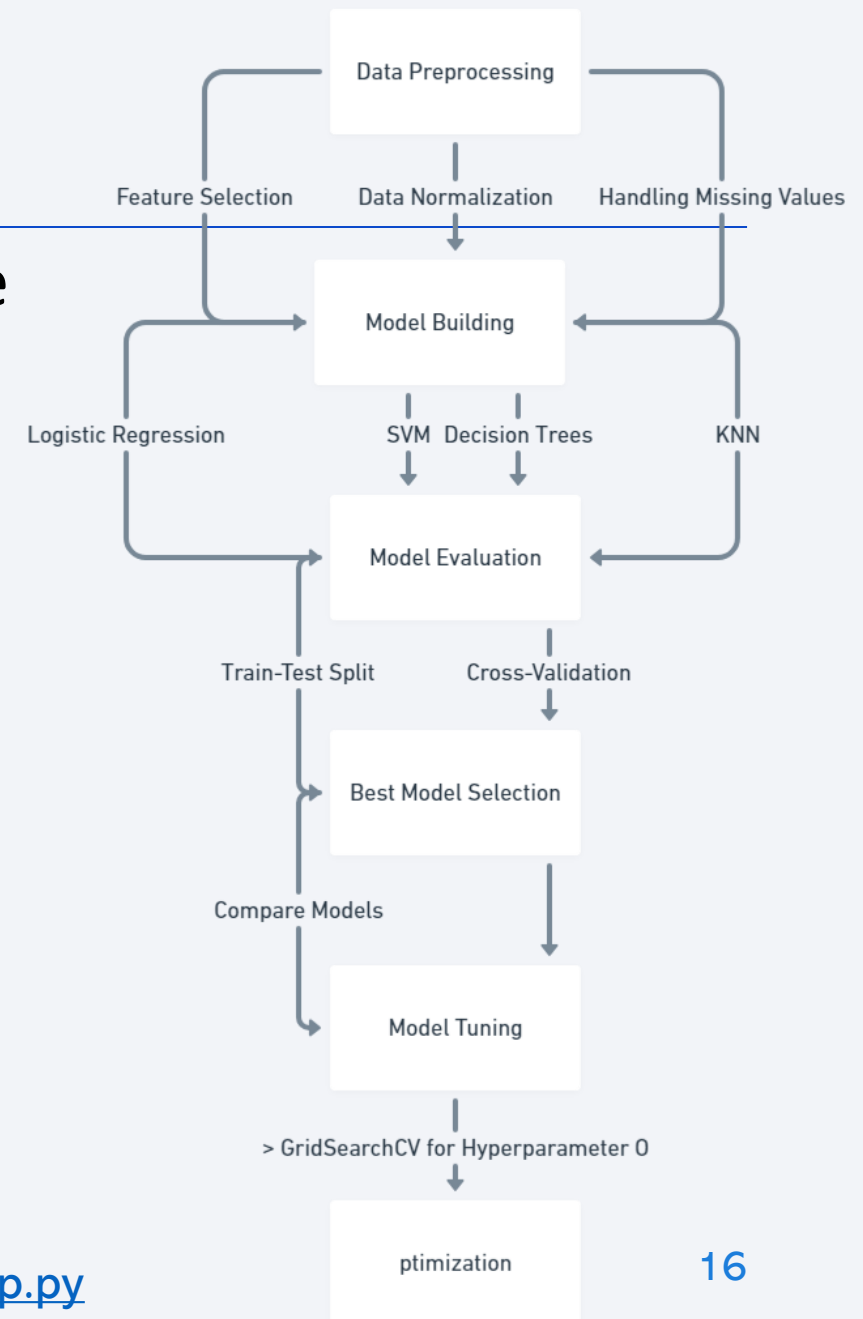
# Predictive Analysis (Classification)

- For predictive analysis the following steps were performed:

- Data Preprocessing (Feature selection, Normalization)
- Model Building (Logistic Regression, SVM, Decision Trees, and KNN)
- Model Evaluation (Train-test split, Cross-validation, Accuracy, Precision, Recall, and F1-score)
- Model Tuning (Hyperparameter optimization using GridSearchCV)
- Model Selection (Compare evaluation metrics, select the best model)

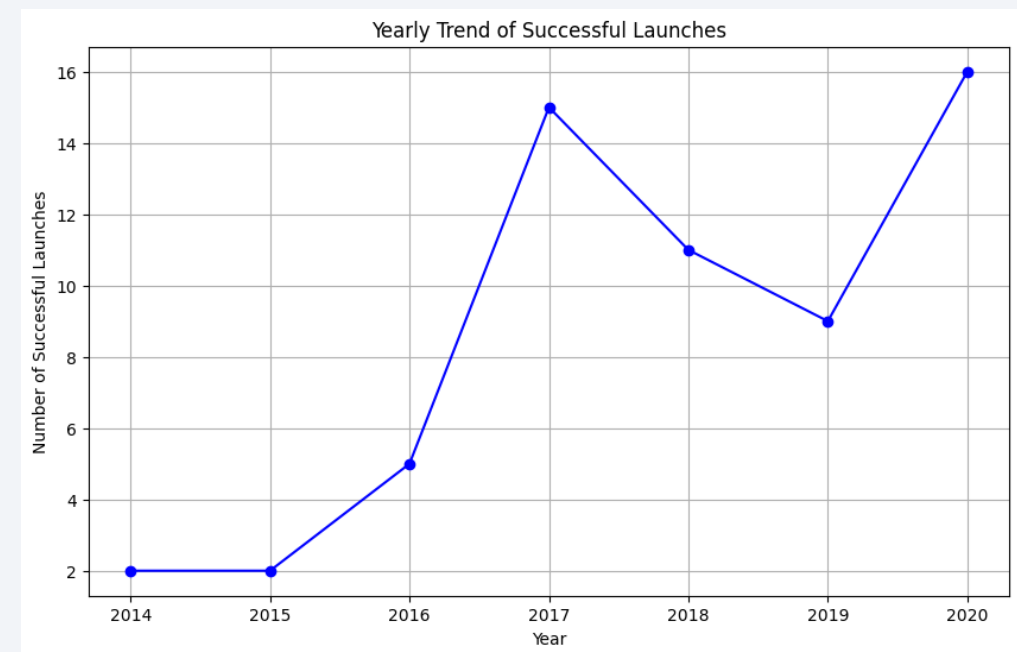
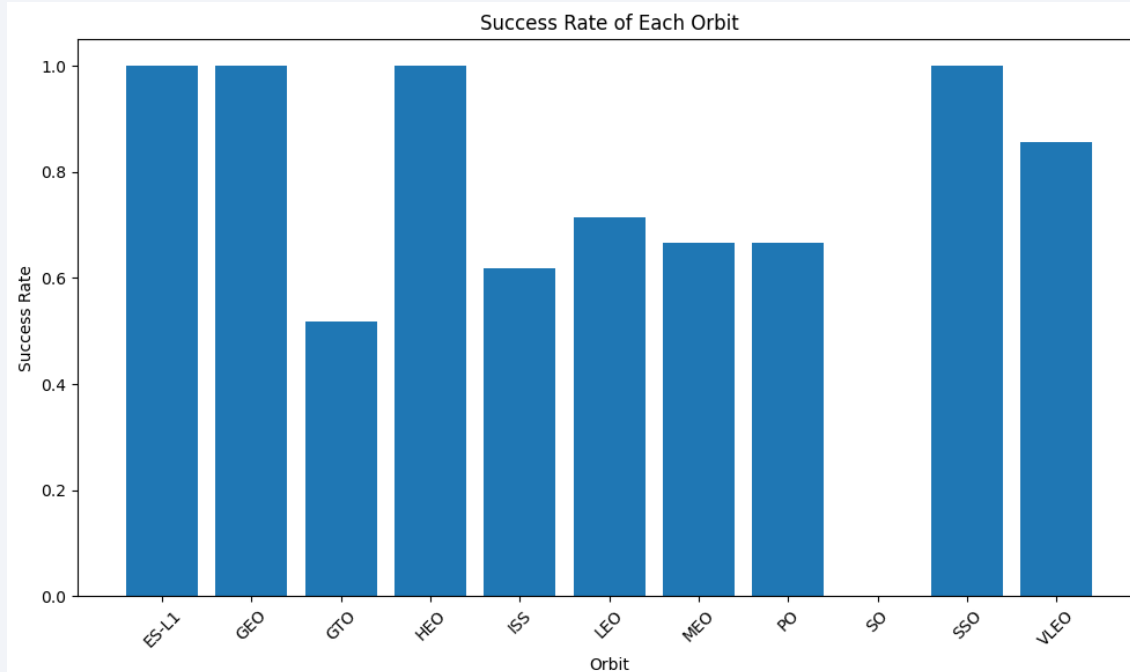
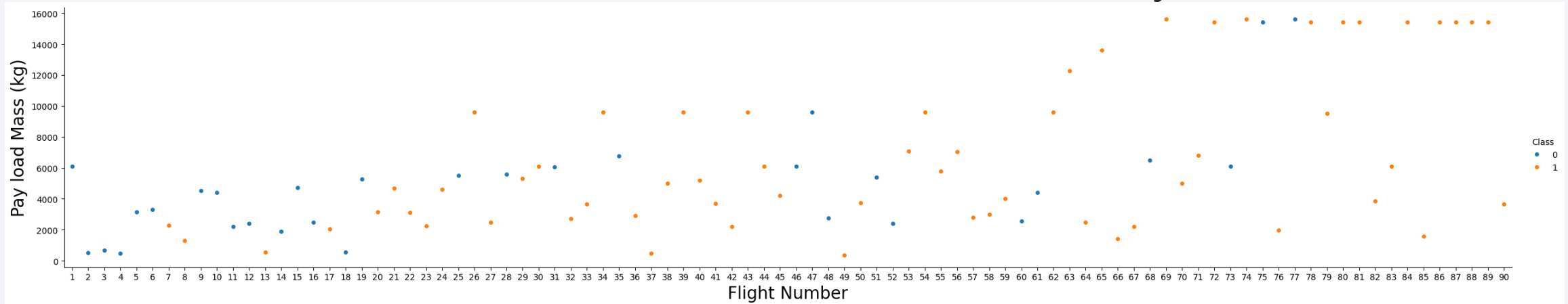
- Predictive Analysis repository is located here:

- [https://github.com/kpcrash/testrepo/blob/main/spacex\\_dash\\_app.py](https://github.com/kpcrash/testrepo/blob/main/spacex_dash_app.py)



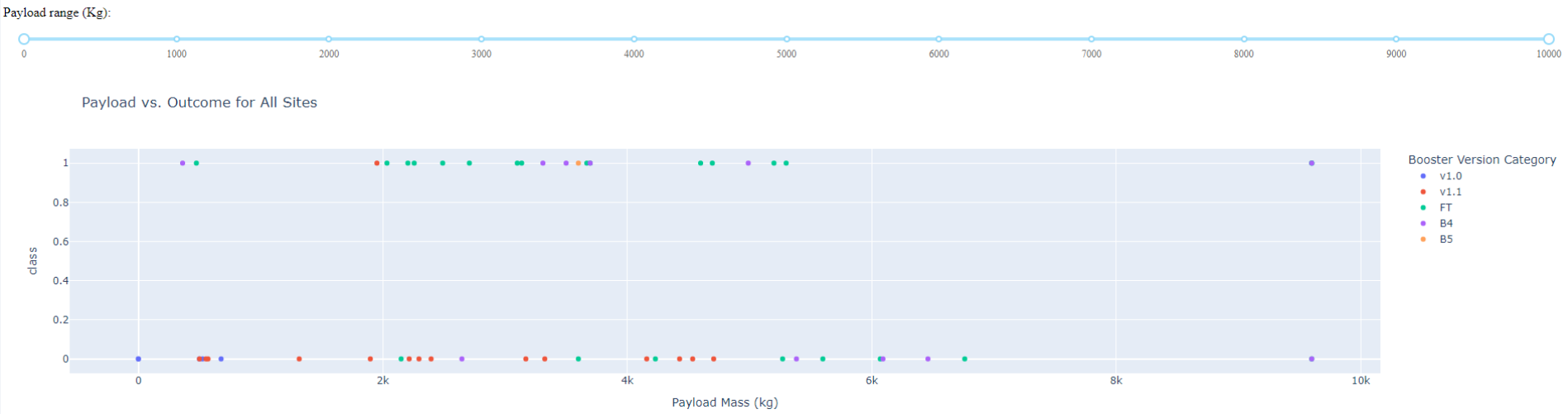
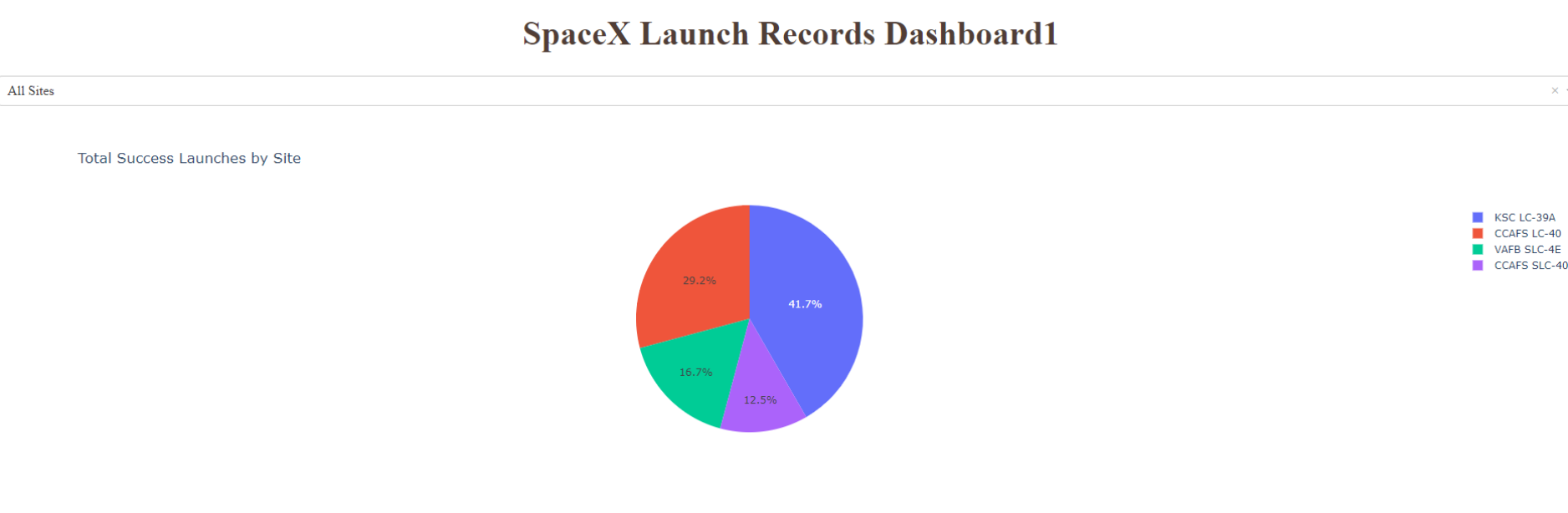
# EDA Results

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



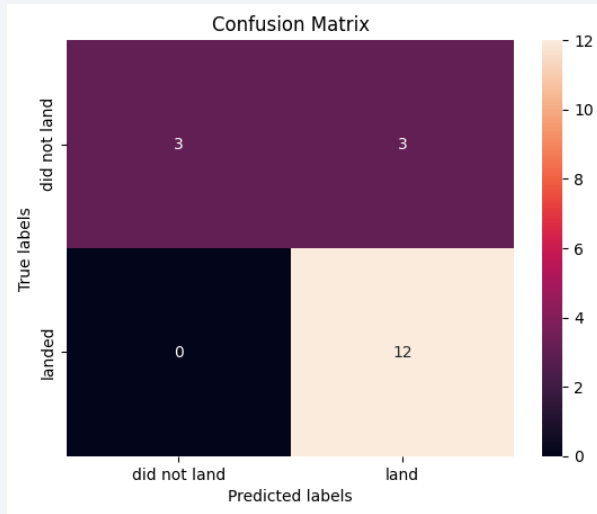
# Interactive Demo Screenshots

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

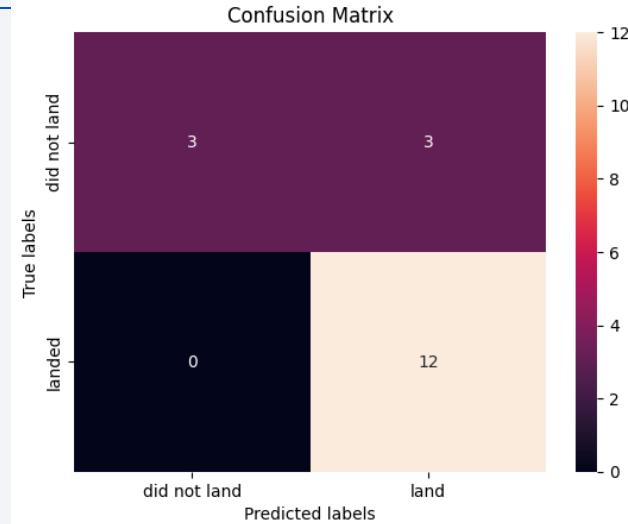




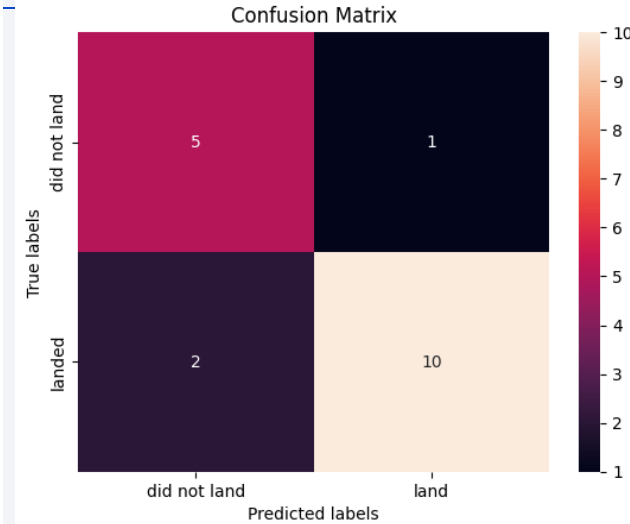
# Predictive Analysis Results



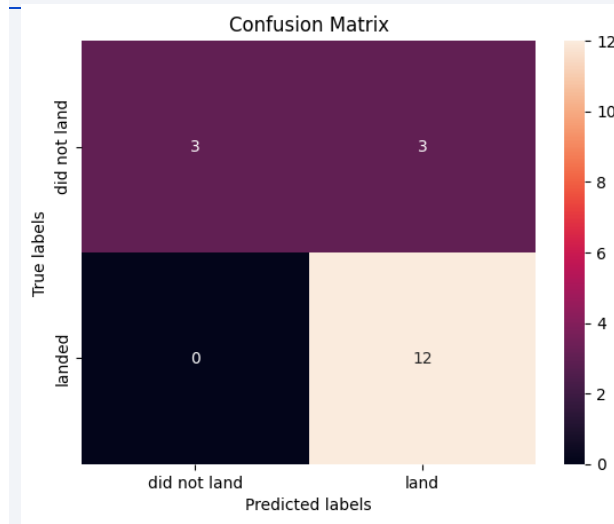
Logistic Regression Results



SVM Results



Decision Tree Results



KNN CV Results

Logistic Regression Test Accuracy: 0.8464285714285713

Support Vector Machine Test Accuracy: 0.8482142857142856

Decision Tree Test Accuracy: 0.8892857142857145

K-Nearest Neighbors Test Accuracy: 0.8482142857142858

**The best performing model is Decision Tree with an accuracy of 0.8892857142857145**



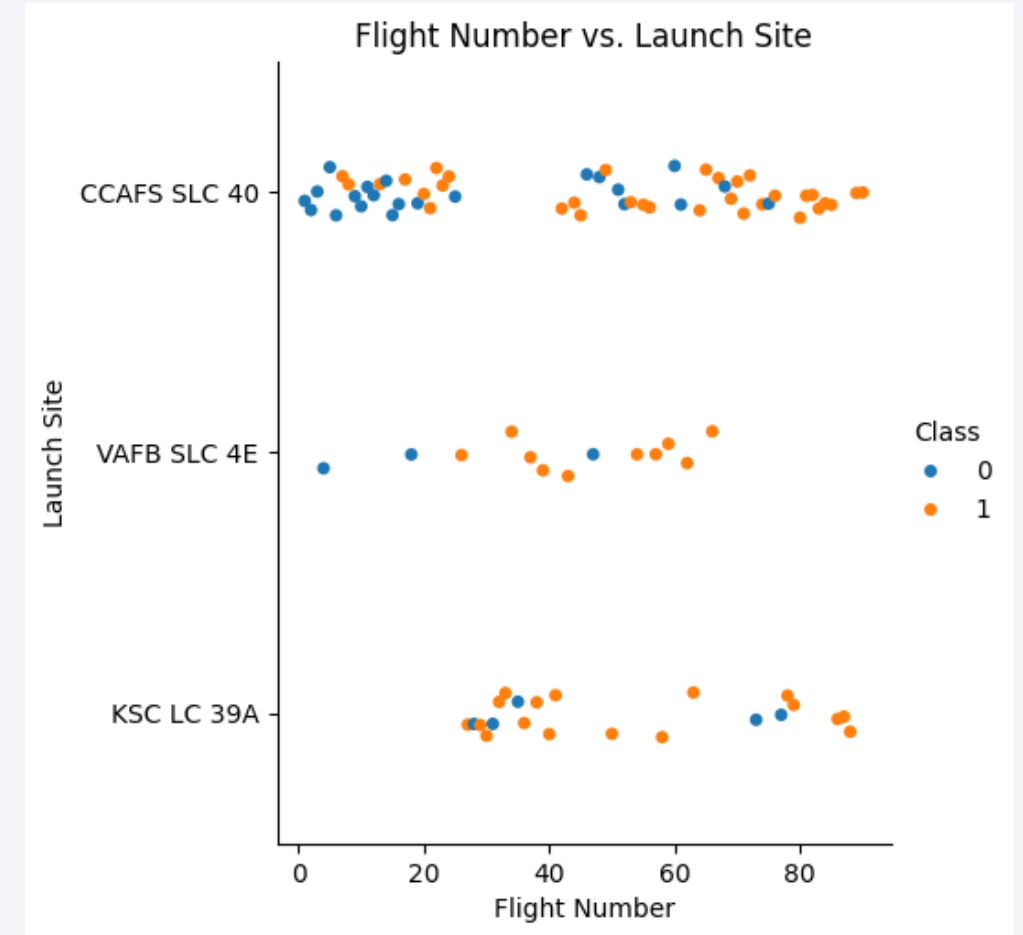
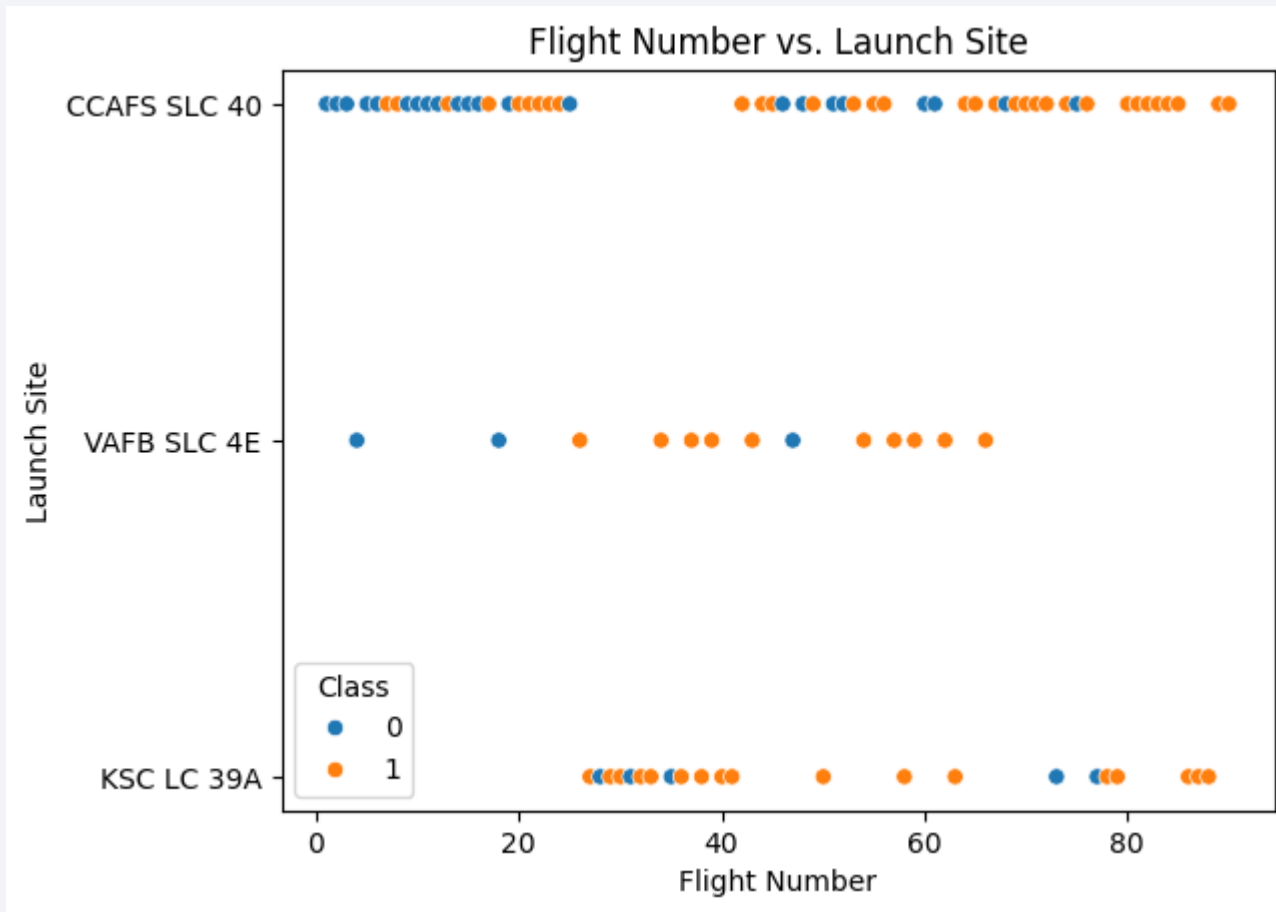
The background of the slide is an abstract composition. It features a dark blue base color. Overlaid on this are numerous diagonal streaks in shades of blue and red, creating a sense of motion or data flow. A faint, light blue grid pattern is also visible, particularly in the lower-left quadrant. The overall effect is high-tech and digital.

Section 2

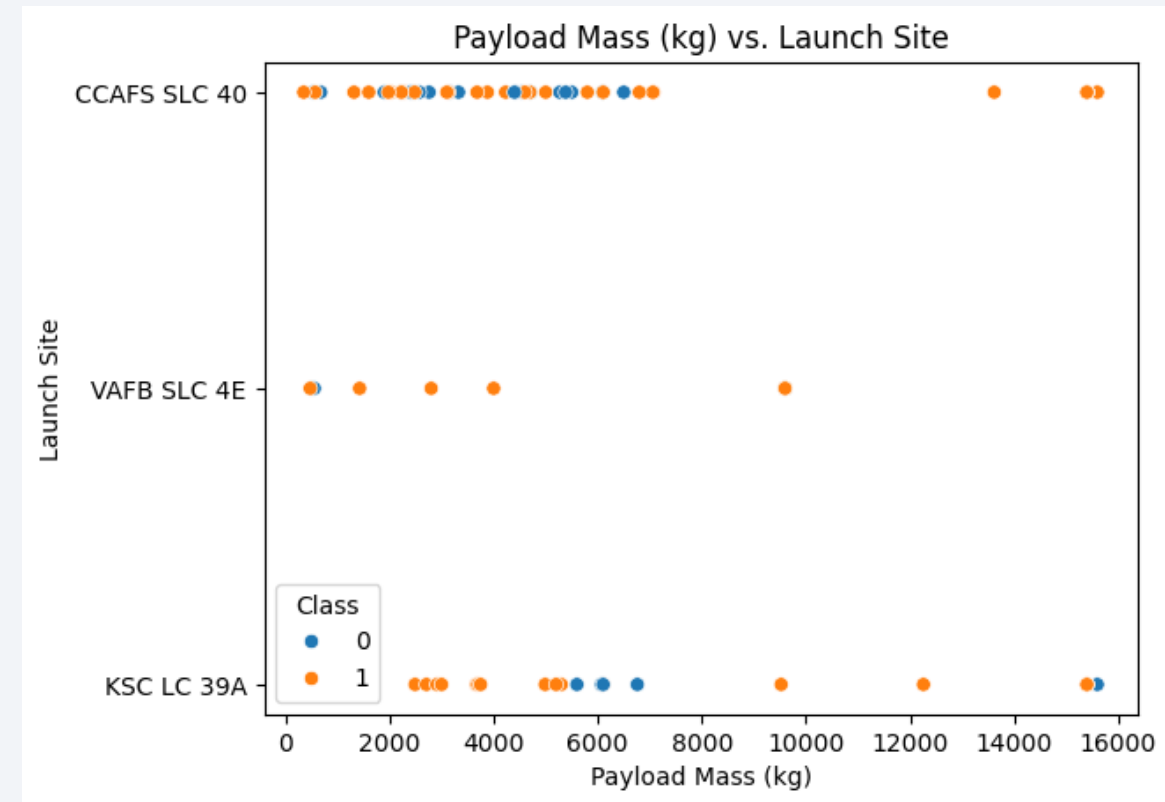
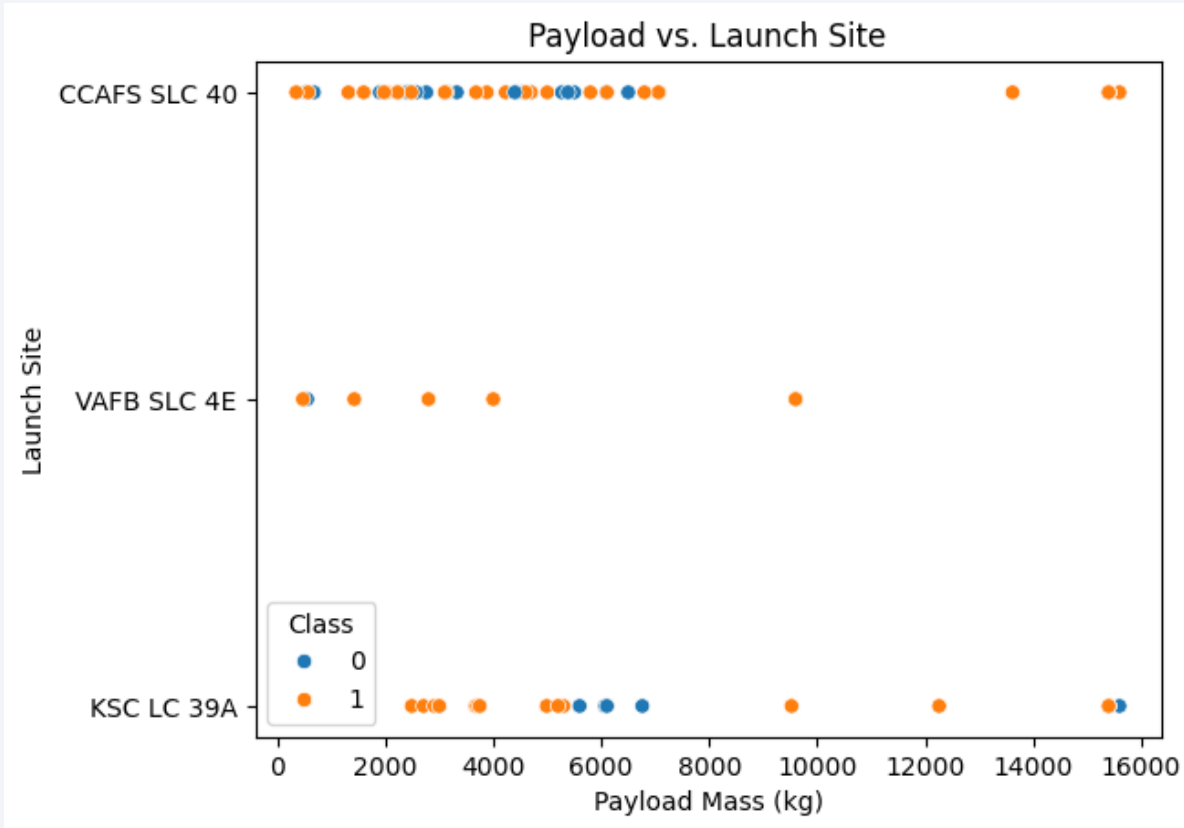
# Insights drawn from EDA



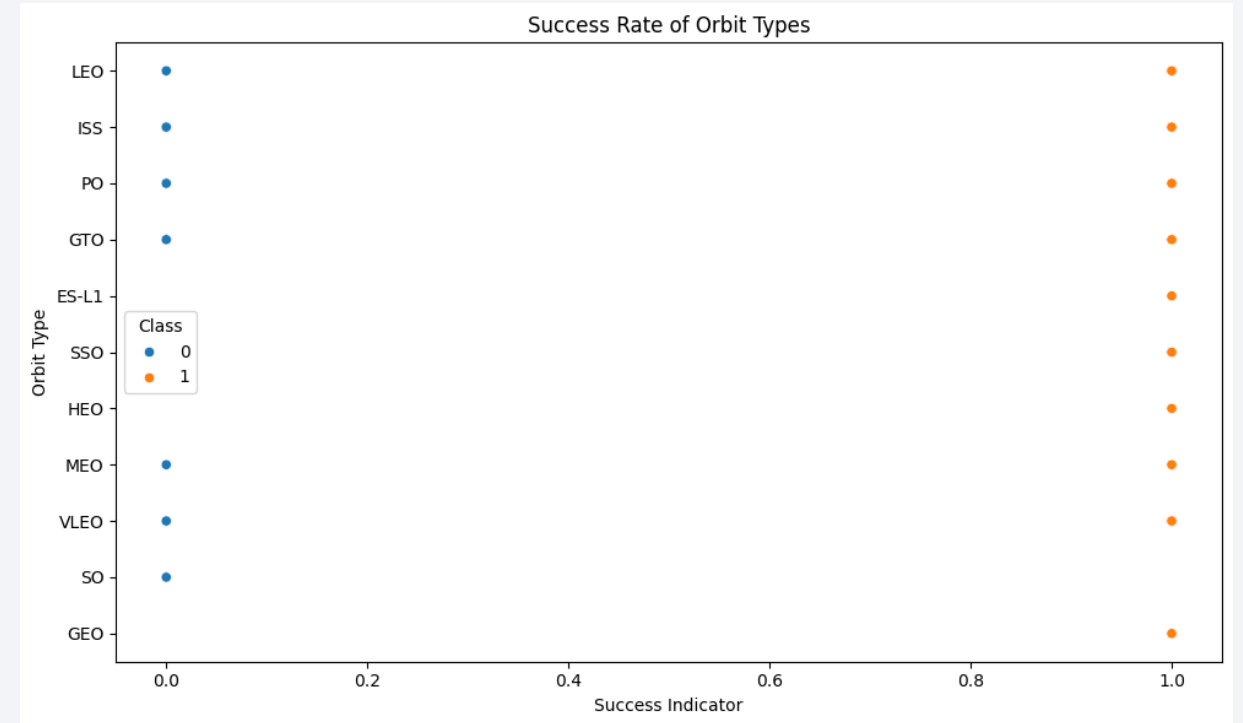
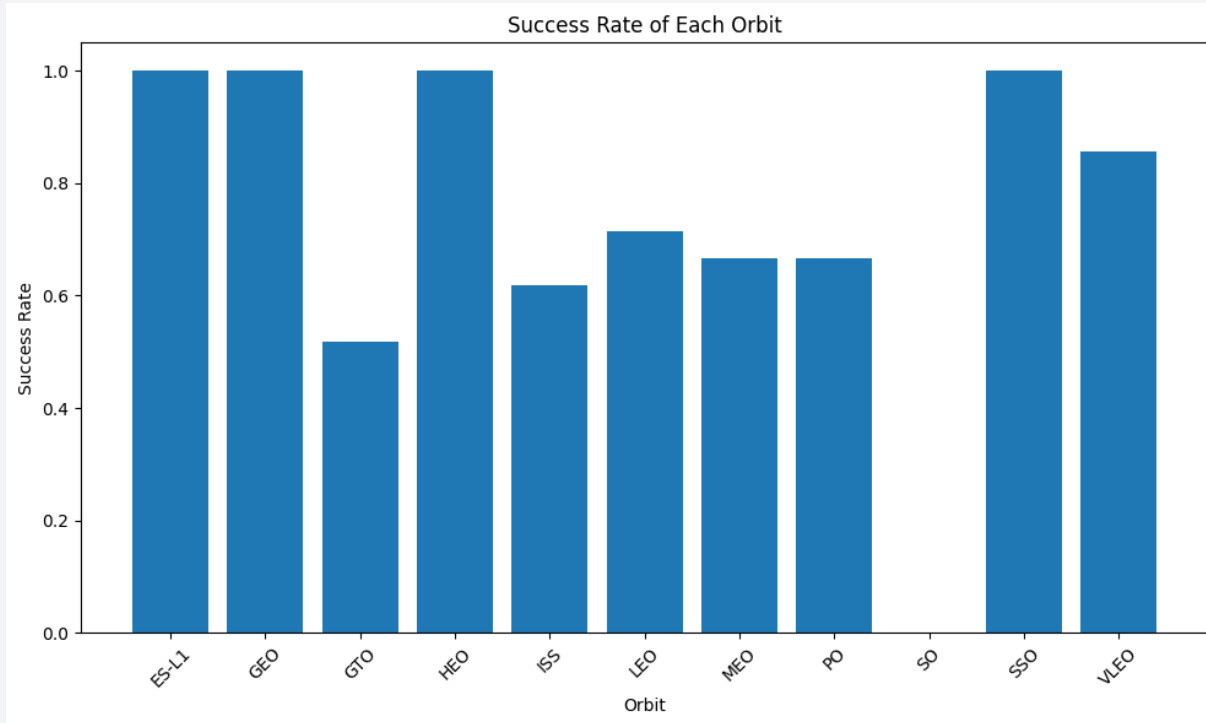
# Flight Number vs. Launch Site



# Payload vs. Launch Site

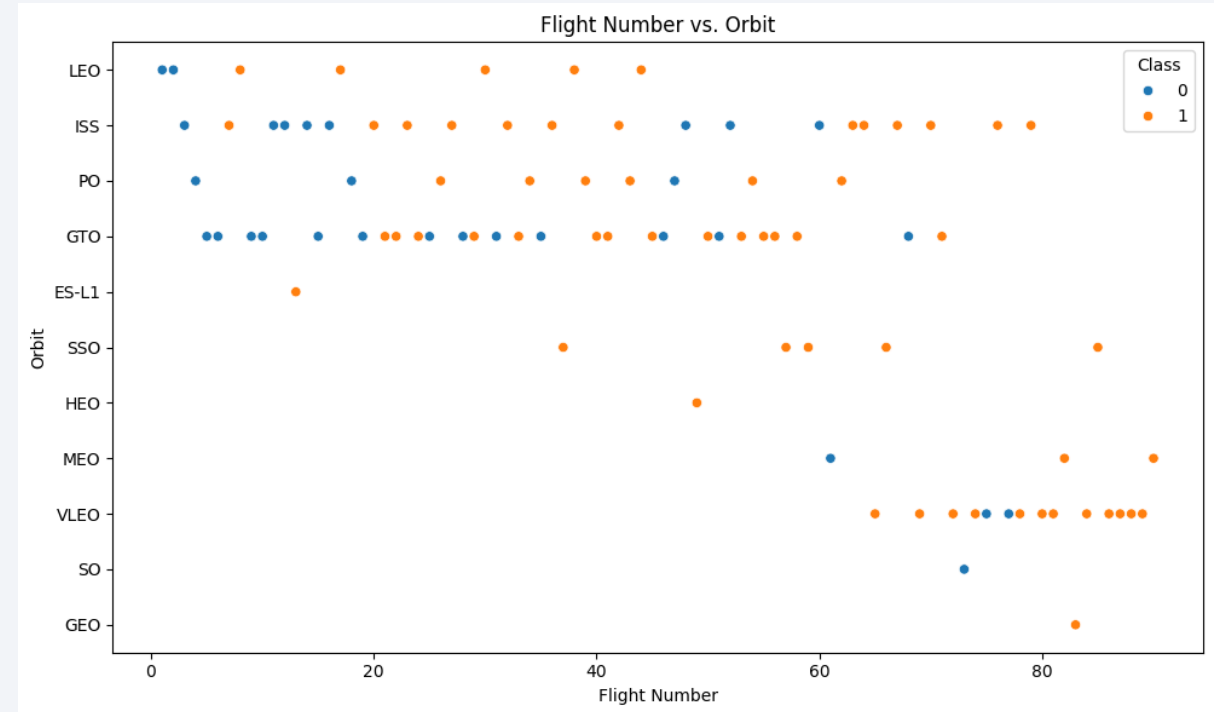
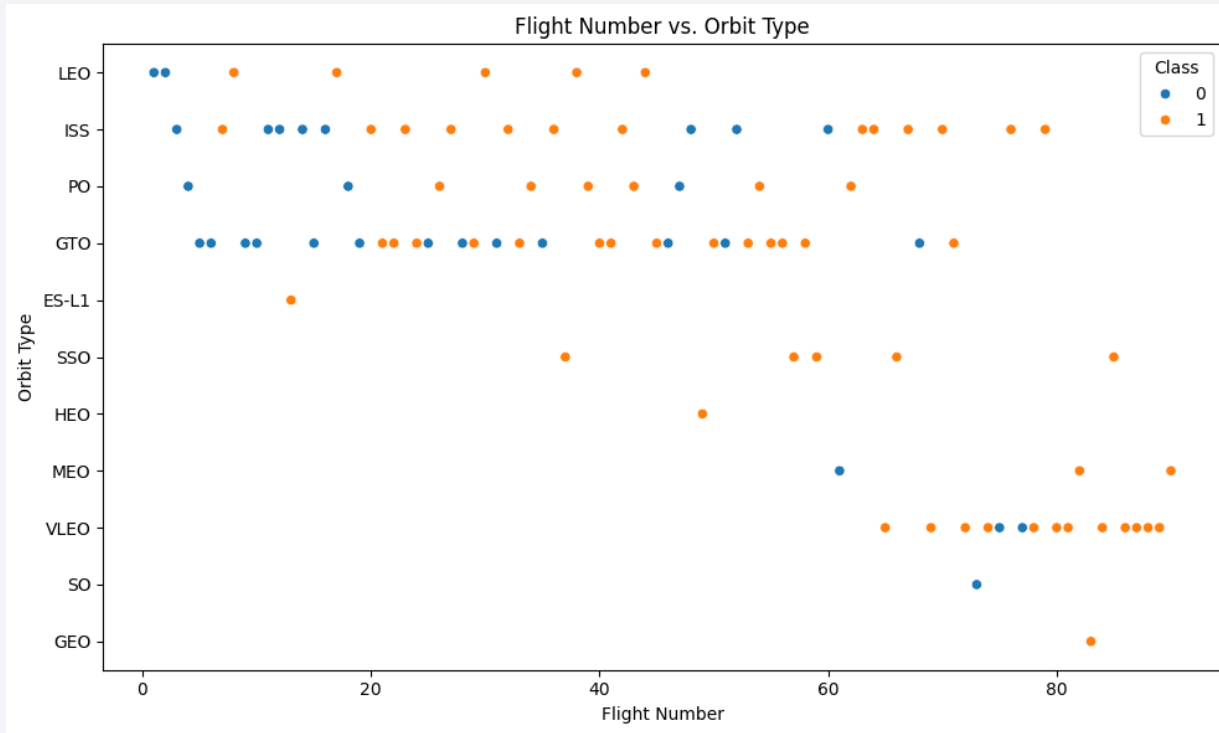


# Success Rate vs. Orbit Type

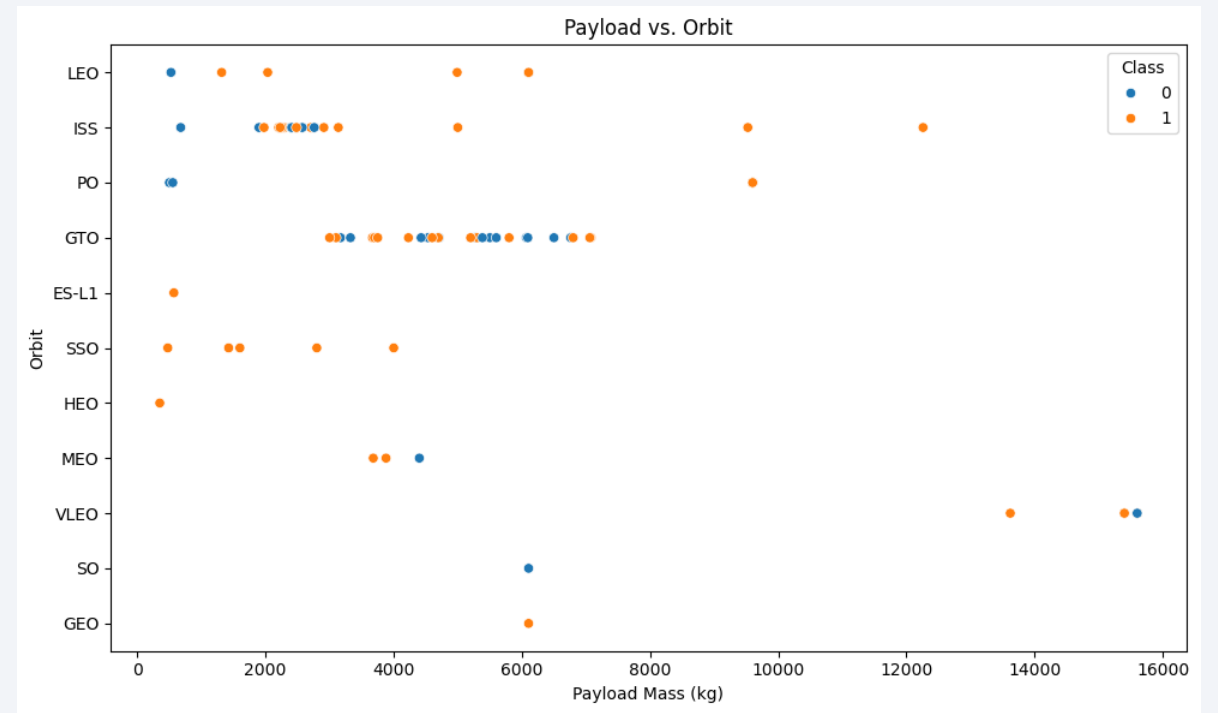
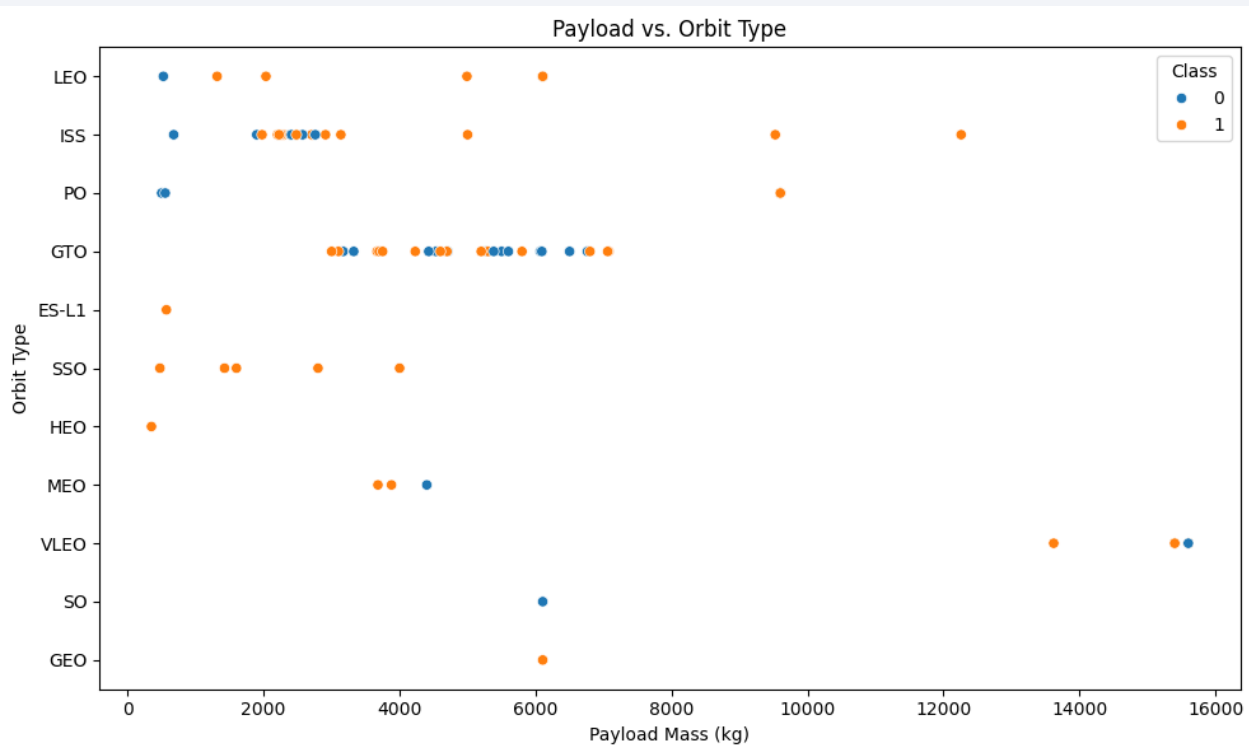




# Flight Number vs. Orbit Type

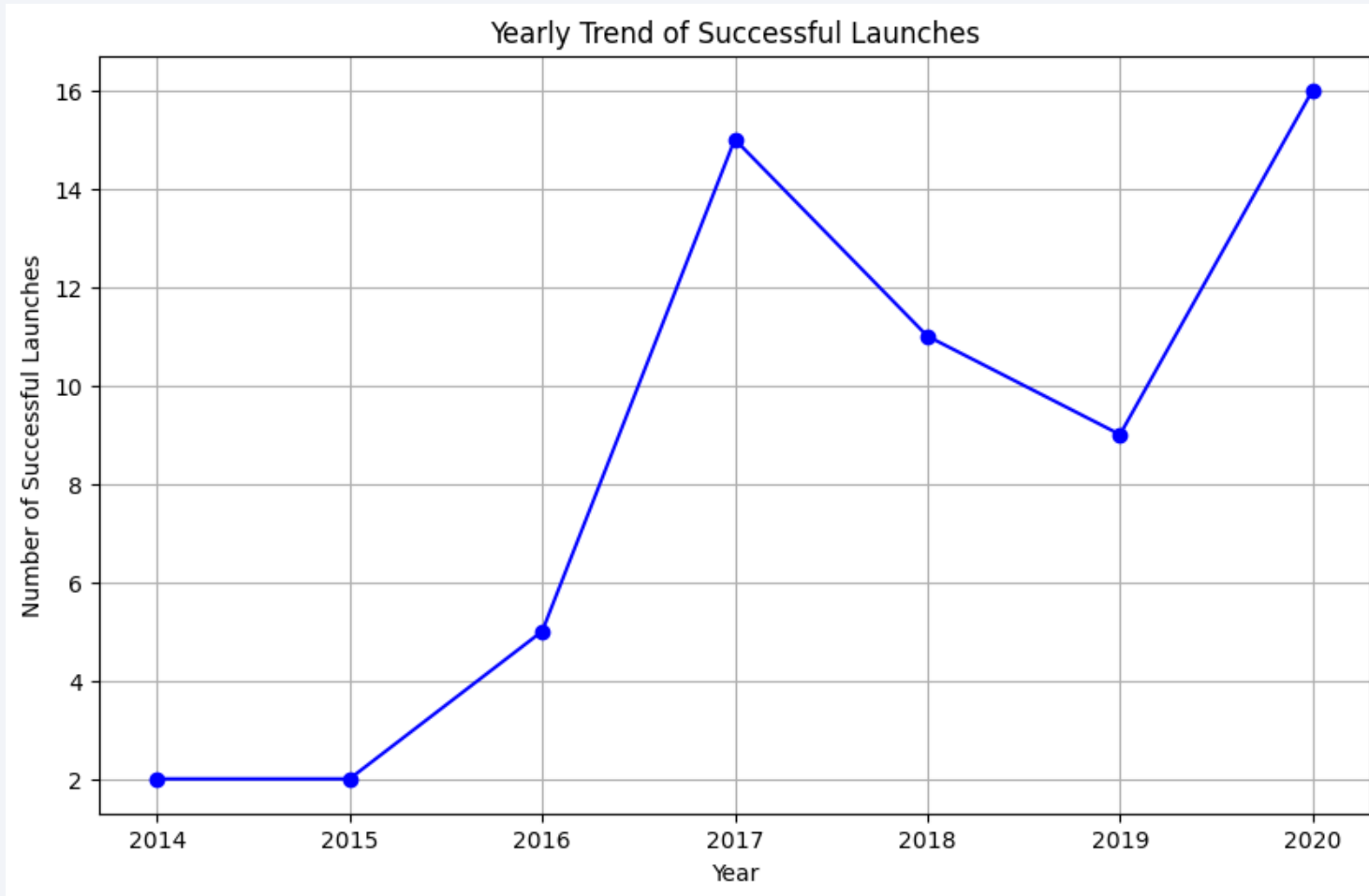


# Payload vs. Orbit Type



# Launch Success Yearly Trend

---



# All Launch Site Names

---

- Find the names of the unique launch sites
- Present your query result with a short explanation here

```
%sql select distinct "Launch_Site" from SPACEXTABLE (Uses distinct to limit  
to the unique names)
```

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

# Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA`
- This query uses the limit function to reduce the results to the first 5 records.

```
[ ] %sql select * from SPACEXTABLE where "Launch_Site" like "CCA%" LIMIT 5
```

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

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



# Total Payload Mass

---

- Calculate the total payload carried by boosters from NASA
- This query uses the SUM function to calculate a total

```
%sql select SUM(PAYLOAD_MASS__KG_) from SPACEXTABLE where Customer = 'NASA (CRS)'
```

```
* sqlite:///my_data1.db  
Done.  
SUM(PAYLOAD_MASS__KG_)  
45596
```

# Average Payload Mass by F9 v1.1

---

- Calculate the average payload mass carried by booster version F9 v1.1
- This query uses the AVG function to calculate an average

```
%sql select AVG(PAYLOAD_MASS_KG_) from SPACEXTABLE where "Booster_Version" = 'F9 v1.1'
```

```
* sqlite:///my_data1.db  
Done.  
AVG(PAYLOAD_MASS_KG_)  
2928.4
```

Task 5

# First Successful Ground Landing Date

---

- Find the dates of the first successful landing outcome on ground pad
- This query uses a where clause and the MIN function to determine the result

```
[ ] %sql select MIN(Date) from SPACE_TABLE where "Mission_Outcome" = "Success"
```

```
↳ * sqlite:///my_data1.db
```

```
Done.
```

```
MIN(Date)
```

```
2010-06-04
```

## 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
- This query uses distinct, a where clause, and a between predicate to get the results

```
[ ] %sql SELECT DISTINCT "Booster_Version" FROM SPACEXTABLE WHERE "Landing_Outcome" LIKE 'Success (drone ship)' AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000;
```



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

```
Done.
```

```
Booster_Version
```

```
F9 FT B1022
```

```
F9 FT B1026
```

```
F9 FT B1021.2
```


```
F9 FT B1031.2
```

# Total Number of Successful and Failure Mission Outcomes

---

- Calculate the total number of successful and failure mission outcomes
- This query combines the SUM function with LIKE clauses and aliasing to determine the result

```
[ ] %sql SELECT SUM(CASE WHEN "Mission_Outcome" LIKE 'Success%' THEN 1 ELSE 0 END) AS Success_Count, SUM(CASE WHEN "Mission_Outcome" LIKE 'Failure%' THEN 1 ELSE 0 END) AS Failure_Count FROM SPACEXTABLE;
```

 \* sqlite:///my\_data1.db  
Done.

Success_Count	Failure_Count
100	1

# Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass
- This query uses a subquery to provide results based on a distinct booster version

List the names of the booster\_versions which have carried the maximum payload mass. Use a subquery

```
[ ] %sql SELECT DISTINCT "Booster_Version" FROM SPACEXTABLE WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE);
```

```
↳ * sqlite:///my_data1.db
```

Done.

**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

# 2015 Launch Records

- List the failed landing\_outcomes in drone ship, their booster versions, and launch site names for in year 2015
- This query requires a custom date function.
- It then uses a where clause with that function to illustrate a result

```
[ ] %%sql
SELECT
    CASE substr(Date, 6, 2)
        WHEN '01' THEN 'January'
        WHEN '02' THEN 'February'
        WHEN '03' THEN 'March'
        WHEN '04' THEN 'April'
        WHEN '05' THEN 'May'
        WHEN '06' THEN 'June'
        WHEN '07' THEN 'July'
        WHEN '08' THEN 'August'
        WHEN '09' THEN 'September'
        WHEN '10' THEN 'October'
        WHEN '11' THEN 'November'
        WHEN '12' THEN 'December'
    END AS Month,
    "Landing_Outcome",
    "Booster_Version",
    "Launch_Site"
FROM SPACEXTABLE
WHERE substr(Date, 1, 4) = '2015'
    AND "Landing_Outcome" LIKE 'Failure (drone ship)';
```



\* sqlite:///my\_data1.db

Done.

Month	Landing_Outcome	Booster_Version	Launch_Site
January	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
April	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40



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

- 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
- This query uses COUNT and RANK functions to produce the results.

```
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
```

```
[25]: %%sql
      SELECT
        "Landing_Outcome",
        COUNT(*) AS Outcome_Count,
        RANK() OVER (ORDER BY COUNT(*) DESC) AS Outcome_Rank
      FROM SPACEXTABLE
      WHERE Date BETWEEN '2010-06-04' AND '2017-03-20'
      GROUP BY Landing_Outcome
      ORDER BY Outcome_Count DESC;
```

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

```
[25]:
```

Landing_Outcome	Outcome_Count	Outcome_Rank
No attempt	10	1
Success (drone ship)	5	2
Failure (drone ship)	5	2
Success (ground pad)	3	4
Controlled (ocean)	3	4
Uncontrolled (ocean)	2	6
Failure (parachute)	2	6
Precluded (drone ship)	1	8

A satellite view of Earth from space, showing the curvature of the planet and city lights at night. The image is a composite of a dark blue sky with stars and a view of the Earth's surface from space. The Earth's surface is mostly dark, with a thin layer of atmosphere visible along the horizon. The city lights are concentrated in the lower right quadrant, showing a dense network of urban areas. The text "Section 3" is overlaid on the left side of the image.

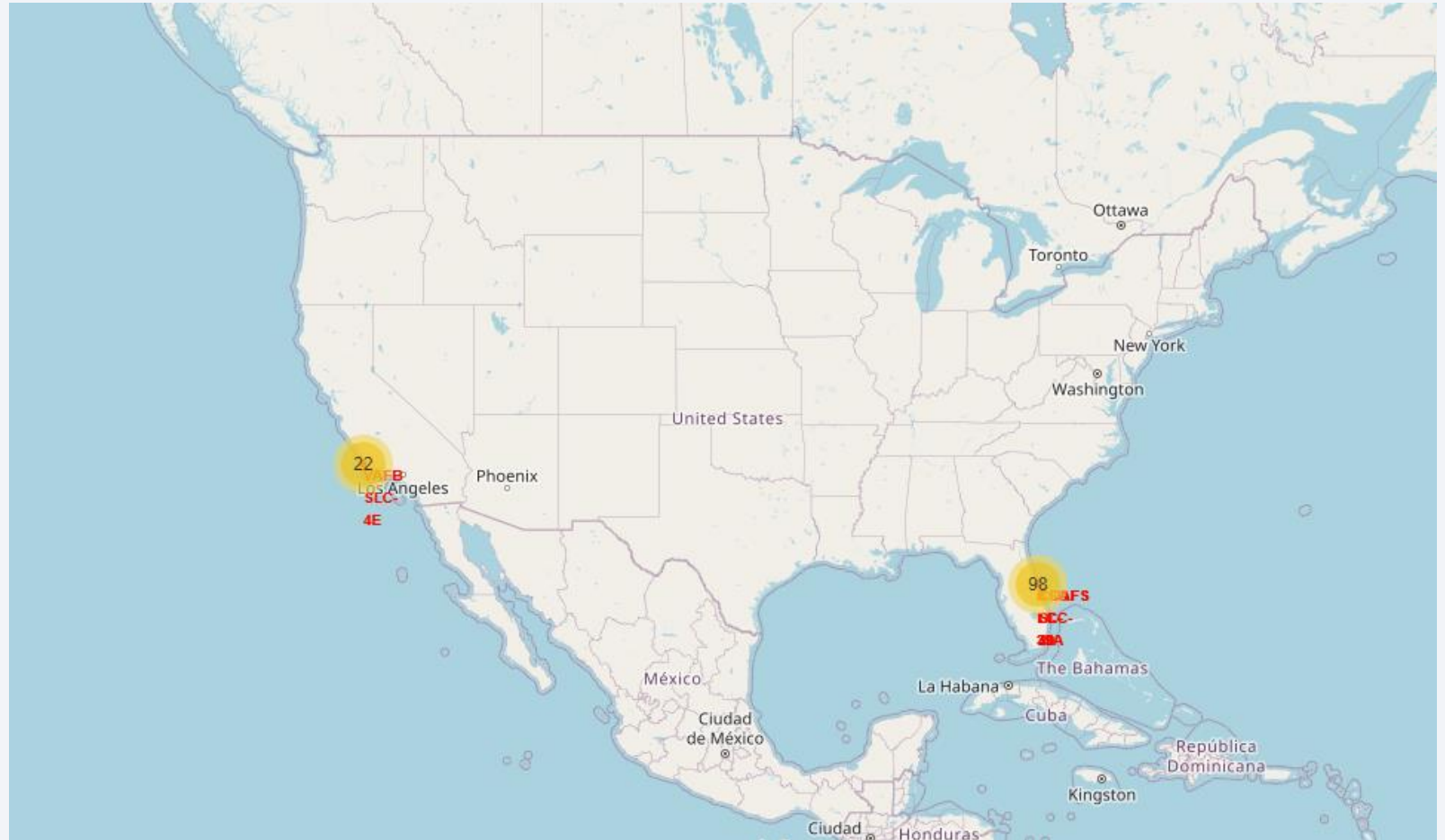
Section 3

# Launch Sites Proximities Analysis

# Launch Site Map

Explain the important elements and findings on the screenshot:

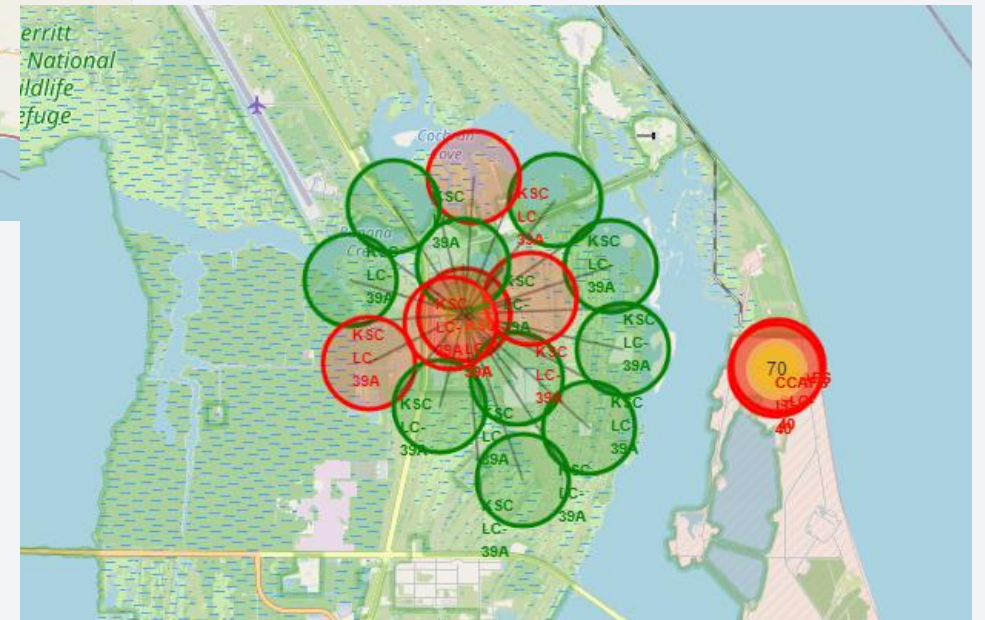
This map shows where all of the SpaceX launch sites are and that they are all located near a coastline.



# Launch Outcomes by Site

Explain the important elements and findings on the screenshot

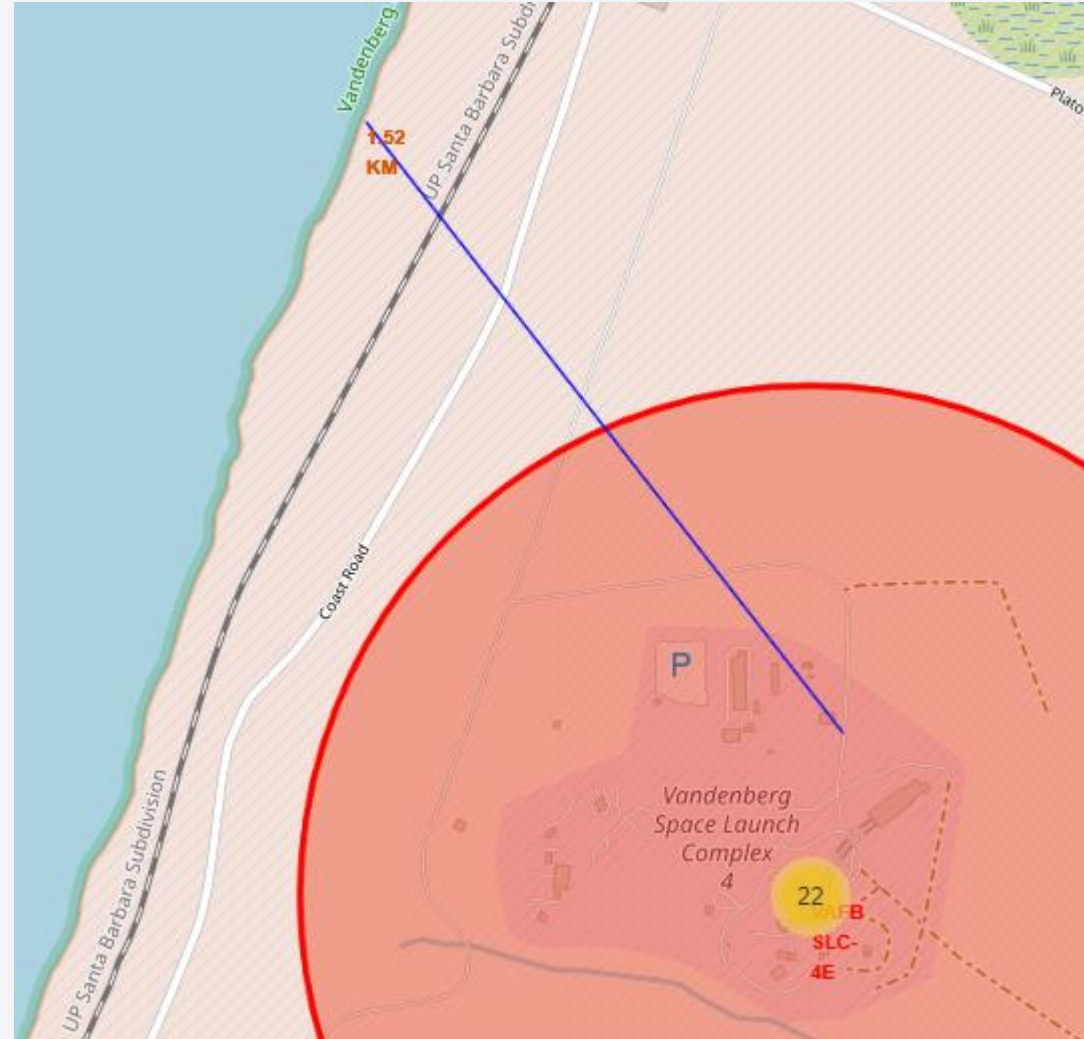
- These images show the success or failure rate at each location as indicated by red (failure) or green (success) markers





# Proximity of Launch Site to Coastline

- This shot shows the proximity of the coastline to the launch site. It also goes over a railroad showing how launch site activities can impact surrounding elements.





Section 4

# Build a Dashboard with Plotly Dash

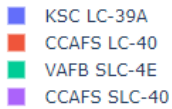
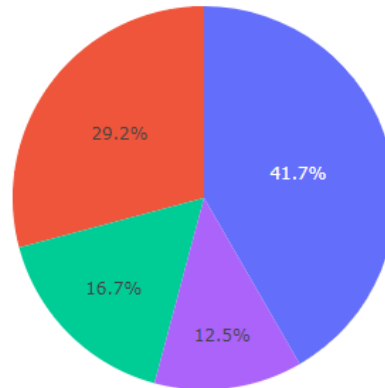


# Successful Launches by Site

---

- This pie chart shows the percentage of total successes by launch site:

Total Success Launches by Site



# Highest Success Ratio for a site

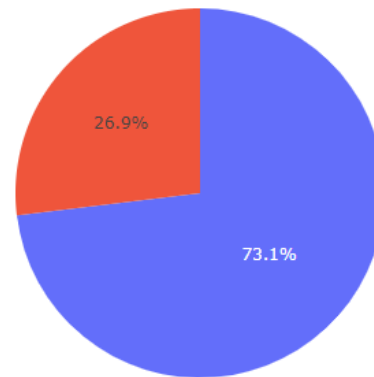
---

- This shows the highest success rate for the sites and that the best success is had at CCAFS LC-40

## SpaceX Launch Records Dashboard1

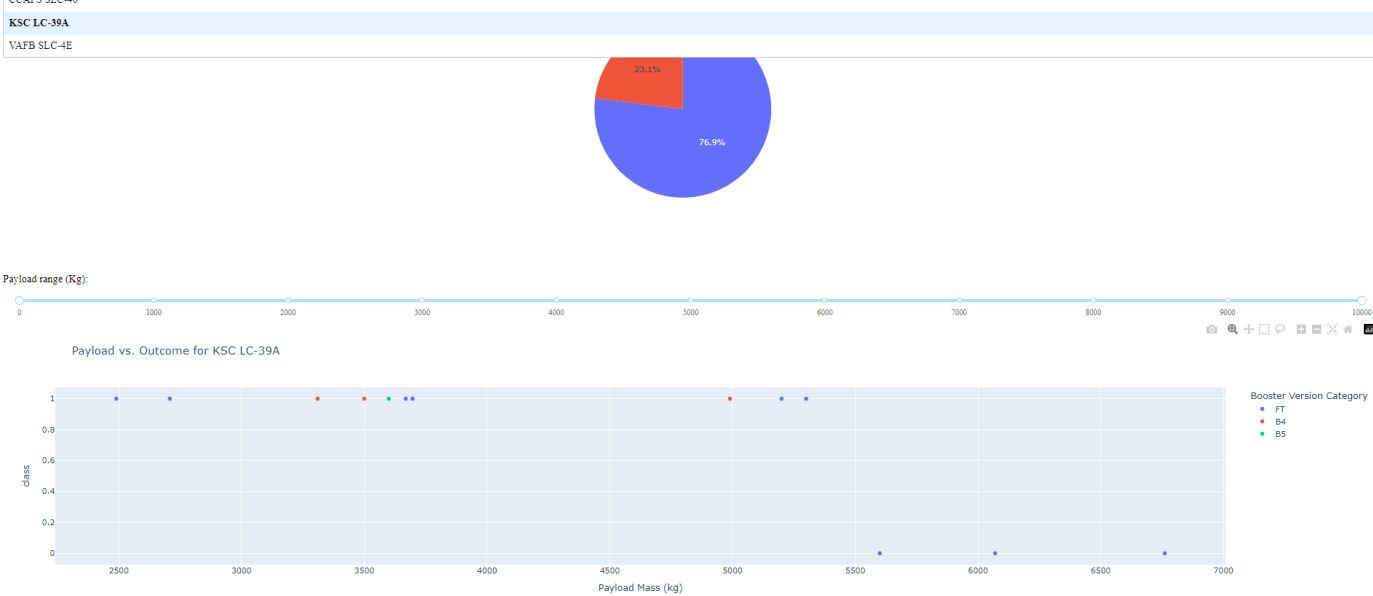
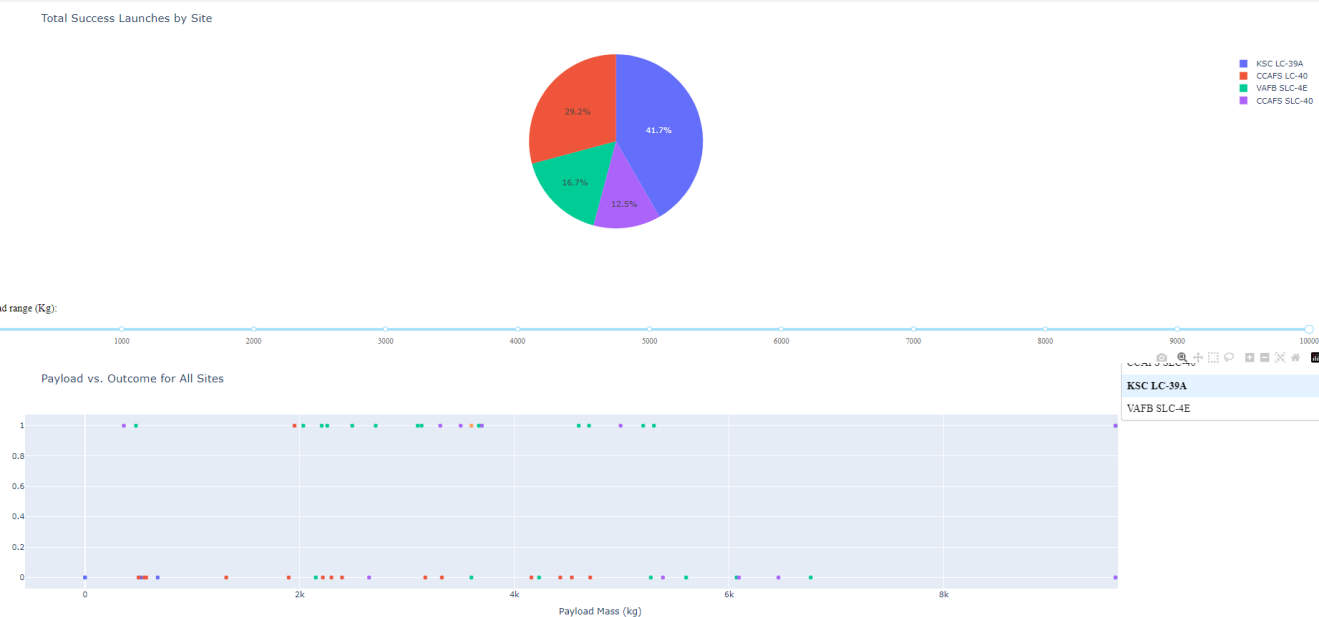
CCAFS LC-40

Total Success Launches for site CCAFS LC-40



# The Impact of Payload on Success

These findings show how the payload impacts the success rate of specific booster versions across sample sites.





Section 5

# Predictive Analysis (Classification)

# Classification Accuracy

---

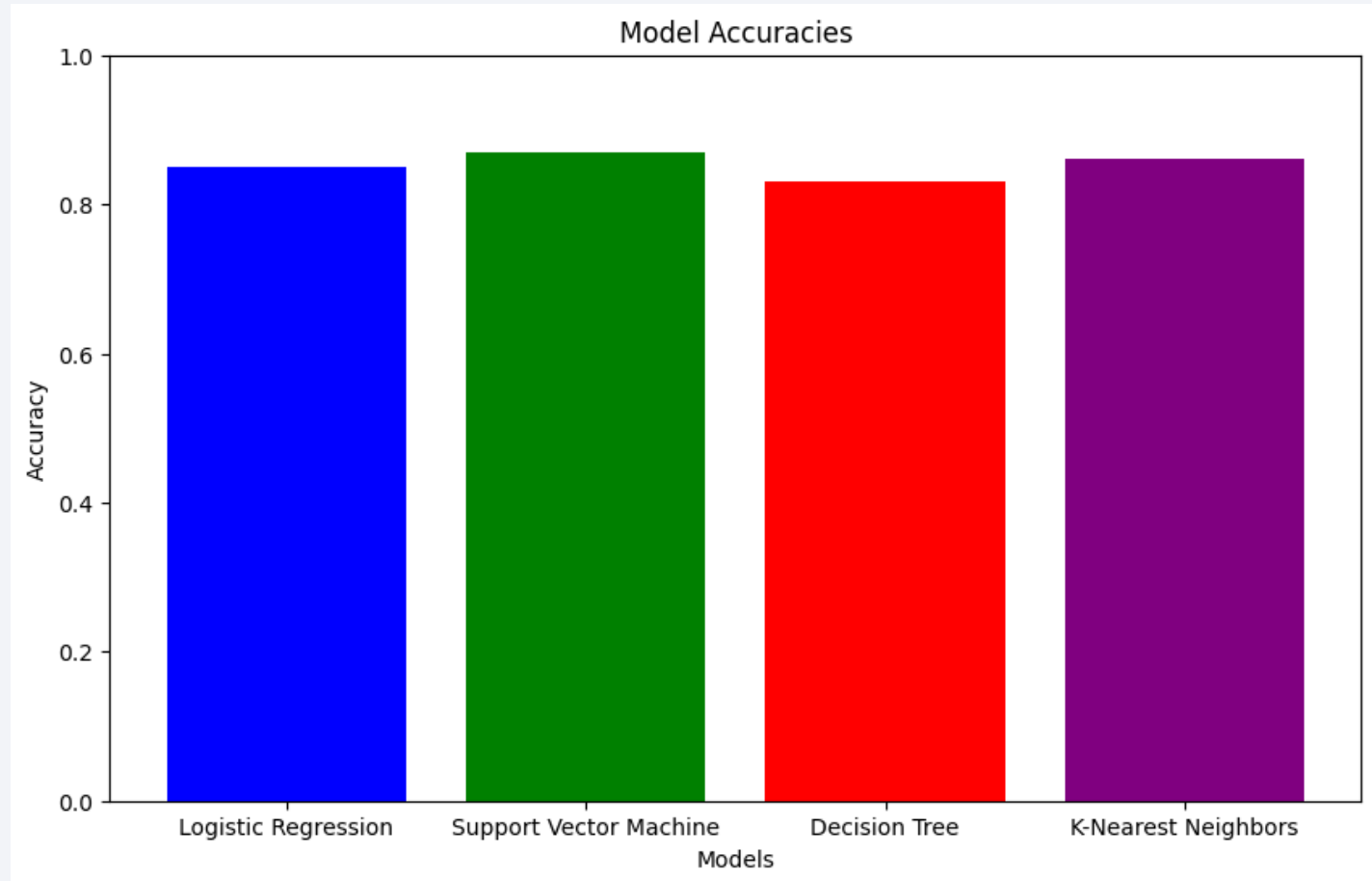
Logistic Regression Test Accuracy:  
0.8464285714285713

Support Vector Machine Test Accuracy:  
0.8482142857142856

Decision Tree Test Accuracy: 0.8875

K-Nearest Neighbors Test Accuracy:  
0.8482142857142858

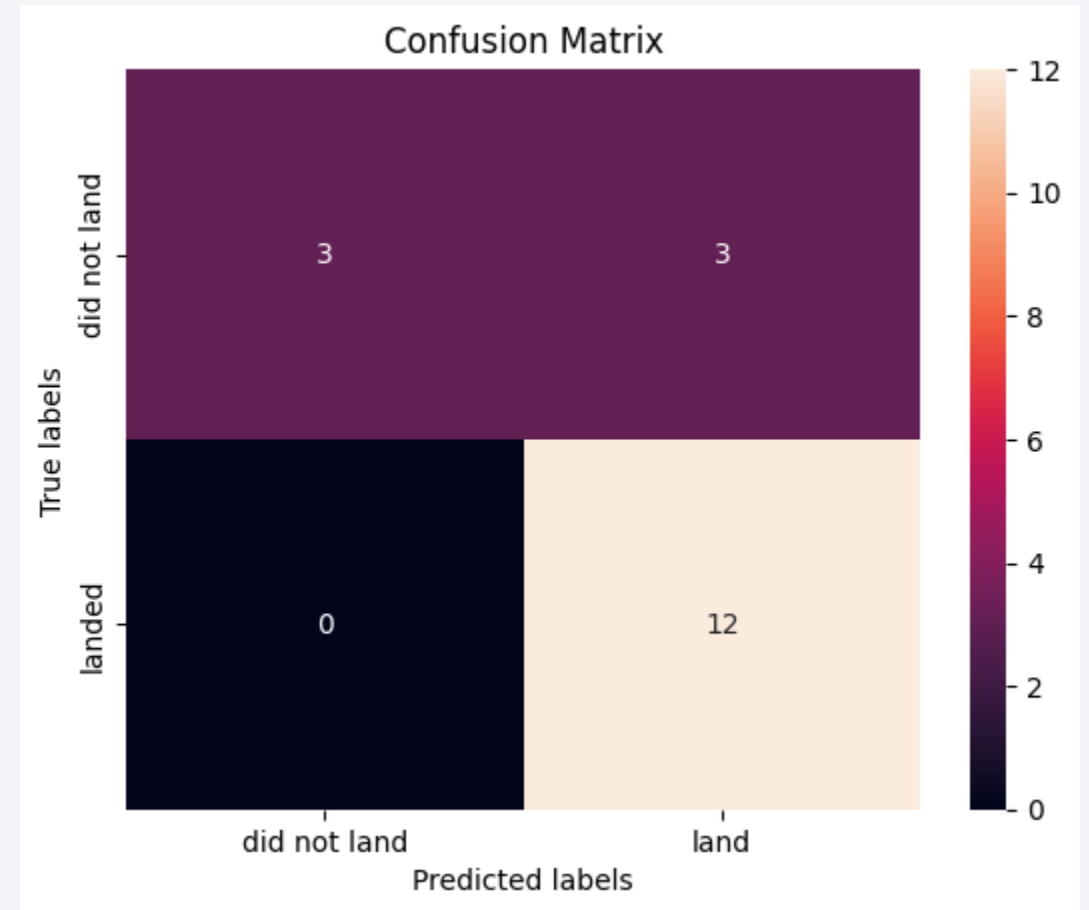
**The best performing model is Decision Tree with an accuracy of 0.8875**



# Confusion Matrix of Best Model

## Components of the Confusion Matrix:

- **True Positives (TP):** Correctly predicted positive instances.
- **True Negatives (TN):** Correctly predicted negative instances.
- **False Positives (FP):** Incorrectly predicted positive instances (Type I error).
- **False Negatives (FN):** Incorrectly predicted negative instances (Type II error).



# Conclusions

---

- We discovered that:
  - Optimizing payload configurations to ensure higher success rates. Heavier payloads might have different success probabilities compared to lighter ones.
  - We need to enhance facilities and processes at launch sites with lower success rates. Different launch sites show varying success rates, likely due to geographical, technical, and logistical factors.
  - We need to invest in booster technologies and refine reuse protocols to maximize launch success. This is because the reuse and type of boosters significantly impact the outcome
- By understanding which factors most affect success rates, SpaceX can allocate resources more effectively, focusing on optimizing crucial aspects such as payload configurations and launch preparations at specific sites.



# Appendix

---

- Please review files in the repository at <https://github.com/kpcrash/testrepo/tree/main> for further information.
- Thank you.

Thank you!

